Pitch-Perfect: A Computer-Simulated Approach

Kuan H. Chen
Chemical Engineering
Applied & Computational Mathematical Sciences

Abstract

In a dystopian society where robots rule the world, artistic expression is nowhere to be found. Robots do not understand art, nor do they need to – it is but a creation appreciated by lowly life-forms, such as humans. This give the human-lead revolutionary army a huge advantage, as messages can be encrypted in various forms of music that robots cannot decipher – or can they? This report discusses the possibility of machines learning how to decipher differences in music, and whether this document is discovered by the robo-cops has a significant impact on the survival of humanity. By using the Fast-Fourier Transformation (FFT), Gabor Transformations (GT), Principle Component Analysis (PCA), and clustering and classification techniques, machine learning can be accomplished, jeopardizing man-kind.

Keywords: FFT, GT, PCA, clustering and classification

Pitch-Perfect: A Computer-Simulated Approach

I. Introduction

Any form of music, on a physical level, is merely a series of waves vibrating at different frequencies, with different shapes, and different amplitudes. This allows one to exploit its underlying mathematical traits, which can be used to decipher between instruments, genres, and even composers. The way this is accomplished is by identifying the dominant characteristics of waves that are shared across all waves produced from a category of music. Such a task is implemented by PCA and a clustering algorithm that will record common wave behavior. Subsequently, any new music data can be compared to the clustering model and be classified into a category. This report first uses image-processing as an example to illustrate how PCA will be implemented, then gives an example of a clustering and classification model, which is shown to be able to decipher between artists and genres.

II. Theoretical Background

1. Fourier Transformation:

The Fourier Transformation is used to decompose a function in the time domain into the frequencies that make it up, the formula for which is shown in Equation 2.

$$\textbf{2. } X(\omega) = \int_{-\infty}^{\infty} x(t)e^{-j\omega t}\,dt$$

After the transformation, the function becomes one that is dependent on frequency instead of time. In the new domain, the magnitude of signals represents the number of detected signals at a given frequency. In this context, it is useful in presenting the frequencies that occur throughout a piece of music.

2. Gabor Transformation:

The GT allows one to analyze data in an enclosed window while ignoring all outside of it. In this context, this window will crop out sections of music for analysis, which, in this report, will be referred to as music clips or clips. The Shannon window is used for signal processing, and is defined as follows in Equation 1,

$$\textbf{1. } \psi_{Shannon}(t) = \begin{cases} 0 \ \forall & t < 0 \\ 1 \ \forall & 0 < t < w \\ 0 \ \forall & w < t \end{cases}$$

where t represents the time frame on which clips are analyzed and w the size of the window.

3. Principle Component Analysis:

The PCA is used to identify the dominant traits in a wave, in this context, the clips. It allows one to simplify a set of data and only retains its key features. Suppose matrix $\boldsymbol{X}$ contains all data of a given system, and that $\boldsymbol{U}$ contains all unitary vectors that form a basis of $\boldsymbol{X}$ (all vectors u_i are **linearly independent**). The matrix $\boldsymbol{Y}$, defined as $\boldsymbol{U^*X}$**,** is a low-rank projection of $\boldsymbol{X}$ onto the basis $\boldsymbol{U}$ containing a set of components that describe $\boldsymbol{X}$. Since any matrix of data $\boldsymbol{A}$ can be decomposed into a collection of component matrices $\boldsymbol{U\Sigma V^T}$, $\boldsymbol{U}$ can be found for $\boldsymbol{X}$. This decomposition is called the Singular-Value Decomposition (SVD) and is shown in Equation 2,

$$\textbf{2.}\, \boldsymbol{A = U\Sigma V^T} = \sum_{\boldsymbol{i=1}}^{\boldsymbol{n}} u_i \sigma_i v_i^T$$

where σ_i represents the standard deviation in the i^{th} dimension. High σ_i suggests high variance, or ${\sigma_i}^2$, in the i^{th} dimension, which in turn represents a strong presence of component u_i in $\boldsymbol{A}$**.** The u_i that corresponds to the highest ${\sigma_i}^2$ is denoted as u_1 and is termed the principle component. The set of u_i that corresponds to high σ_i can be used to find $\boldsymbol{Y}$. Equation 3 can be used as a measure to determine whether an appropriate $\boldsymbol{Y}$ is found.

$$\mathbf{3.}\, f = \sum_{i}^{m} {\sigma_i}^2 \Big/ \sum_{i}^{n} {\sigma_i}^2 \,, m < n$$

When the variation in f is little as i increases, the corresponding matrix is $\boldsymbol{Y}$.

4. **Clustering and Classification:**

Many clustering and classification algorithms are available for separating data apart into clusters. These clusters provide a basis that is used to classify additional information that is received. In this report, the Discriminant Analysis Classifier (DAC) is used to cluster the principle components of clips, which can be used to classify clips that are subsequently introduced to the algorithm. The DAC uses Gaussian curves, to identify and separate clusters.

III. Algorithm Implementation and Development

1. **Yale Images:**

A series of cropped images of faces are provided as an exercise to illustrate how PCA may be implemented. Sample images are shown in Figure 1.

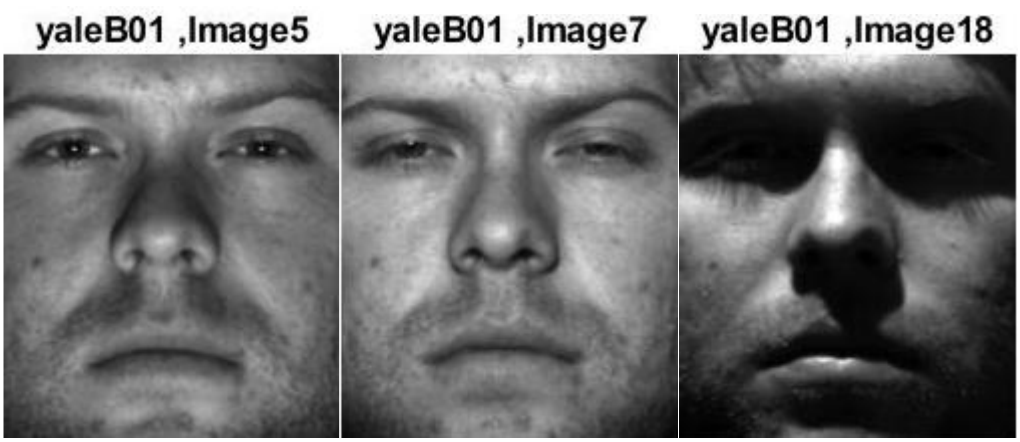

Figure 1. Sample Images

These can be seen simply as waves in 3-D with its amplitude dependent on the spatial components defined in Euclidean space. In order to identify and record the commonalities between each image, the image data must be collected to perform a PCA analysis. This can be accomplished by reshaping each image into a vector and concatenating them side-by-side to form the matrix $\boldsymbol{A}$ as discussed in Equation 2. This is accomplished using the following MATLAB commands:

```
ScroppedYale = dir(fullfile(DcroppedYale,folderName,'*.pgm'));
for k = 1:numel(ScroppedYale)
    F = fullfile(DcroppedYale,folderName,ScroppedYale(k).name);
    I = imread(F);
    Ireshaped = reshape(I,numel(I),1);
    reshapedImages = [reshapedImages, Ireshaped];
```

Next, the principle components of this matrix $\boldsymbol{A}$ are found by performing the svd on it, which is accomplished by the following MATLAB command:

```
[U,S,V] = svd(double(reshapedImages),'econ');
```

Where U contains the principle components of $\boldsymbol{A}$ and S contains the variances of the principle components. Figure 2 shows how the variances change with its principle component.

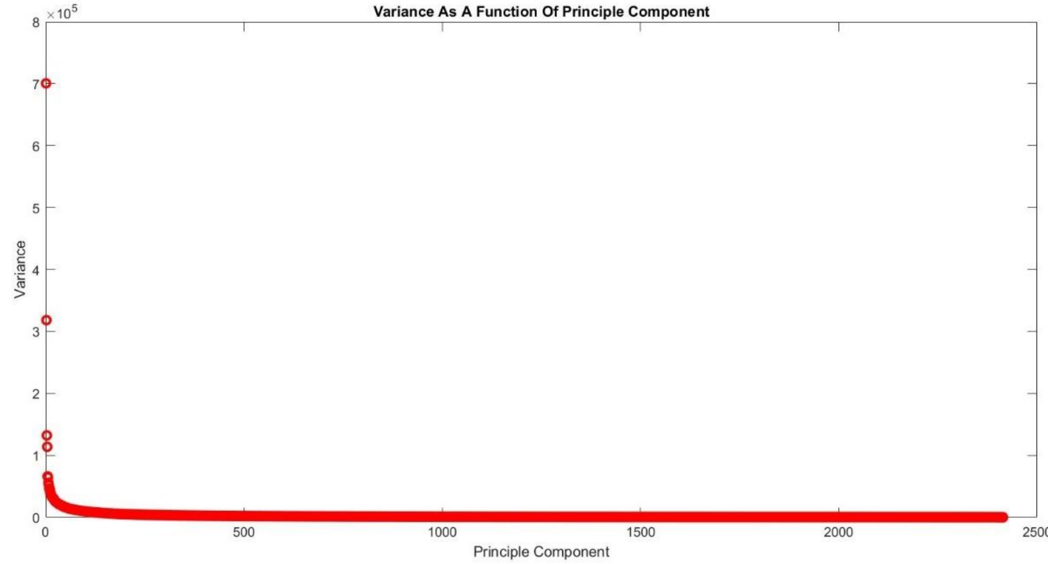

Figure 2. Variance as a function of principle component for cropped images

Figure 2 suggests that few, perhaps 2, principle components would be needed to retain sufficient information to describe these. This is expected, given that the images are all cropped and in focus. A similar procedure as described above can be used to analyze a series of uncropped images, the variances for which as a function of principle component is shown in Figure 3.

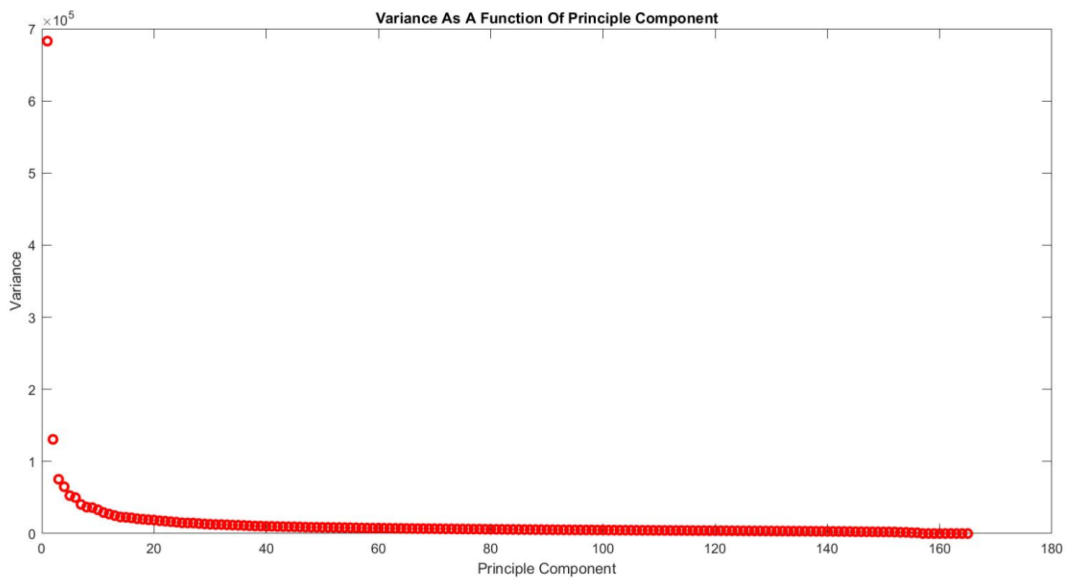

Figure 3. Variance as a function of principle component for uncropped images

2. Music Classification:

In order to build a clustering model for clip analysis, clips need to be collected first. It should be noted that the author is deliberately using this as a platform to promote his favorite band of music, so the reader is advised to look up recommended pieces that are presented here. The author's personal bias for classical music has prompted him to categorize music into the following three categories – transcendental (Classical), generally tolerable (Pop), and vulgar (Rap). Pieces of music are selected for these categories to perform analysis.

First, a clustering model that separates music based on artists of different genres is implemented (Test 1). Music from Lepold Godowsky (the best piano composer in all of history!), Camila Cabello, and the Black-Eyed Peas (BEP) are sampled. In order to retain the time information of these pieces, GT's are first performed on each piece. The Shannon window is used to crop music data into clips of 5 seconds. Spectrograms of these clips are then taken to record their information in the Fourier domain and concatenated with each other (similar to the case in the Yale Images section) to form the matrix $\boldsymbol{A}$. These can be implemented with the following MATLAB commands:

```
allMusicData = [];
for t = 0:samplingTime:(numSamples-1)*samplingTime
    ii = t + 1;
    musicClip = musicFile(Fs*ii:Fs*(ii+samplingTime),1);
    musicClip = abs(spectrogram(musicClip));
    musicClip = reshape(musicClip,[numel(musicClip), 1]);
    allMusicData = [allMusicData musicClip];
end
```

Where `samplingTime` is 5. Figure 4 shows the variance of $\boldsymbol{A}$ as a function of principle component.

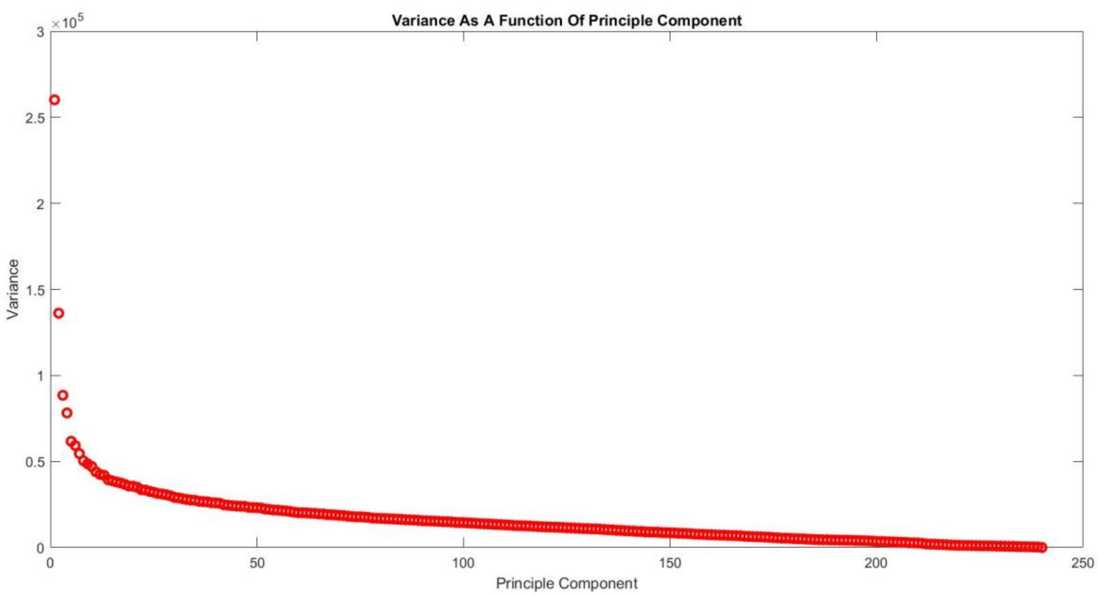

Figure 4. Variance as a function of principle component for the clips

This suggests 2 of the principle components are sufficient in representing all music data.

Figure 5 shows how the first two principle components of the clips from each category are distributed if represented as data points on a plane.

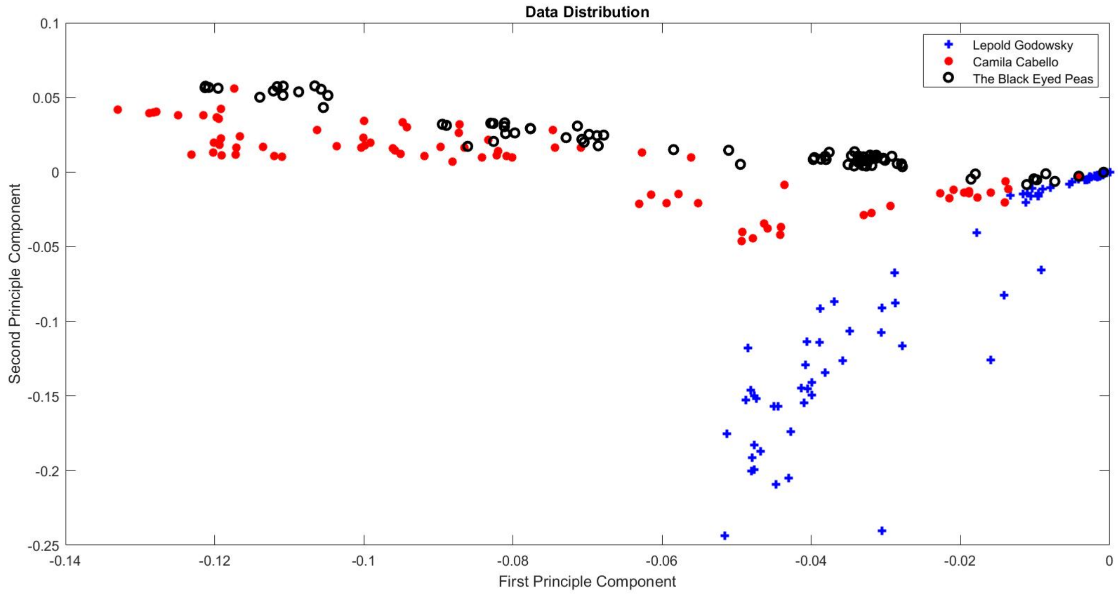

Figure 5. Data distributions between Godowsky (Classical), Cabello (Pop), and the BEP (Rap)

From this plot, it can be seen that "God"owsky's music, being as Godly as it is, exhibits separation from the other two set of data; Cabello's and the BEP's music exhibit overlaps, but are somewhat separated. These observations will later be exploited.

Similar distribution plots can be generated for the cases in which clustering models based on artists of the same genre (Test 2) and several artists from different genres (Test 3) are implemented. Figure 6 shows the distribution plot generated using music from Chopin, Godowsky, and Rachmaninoff (all classical); Figure 7 shows the distribution plot generated using music from classical, pop, and rap.

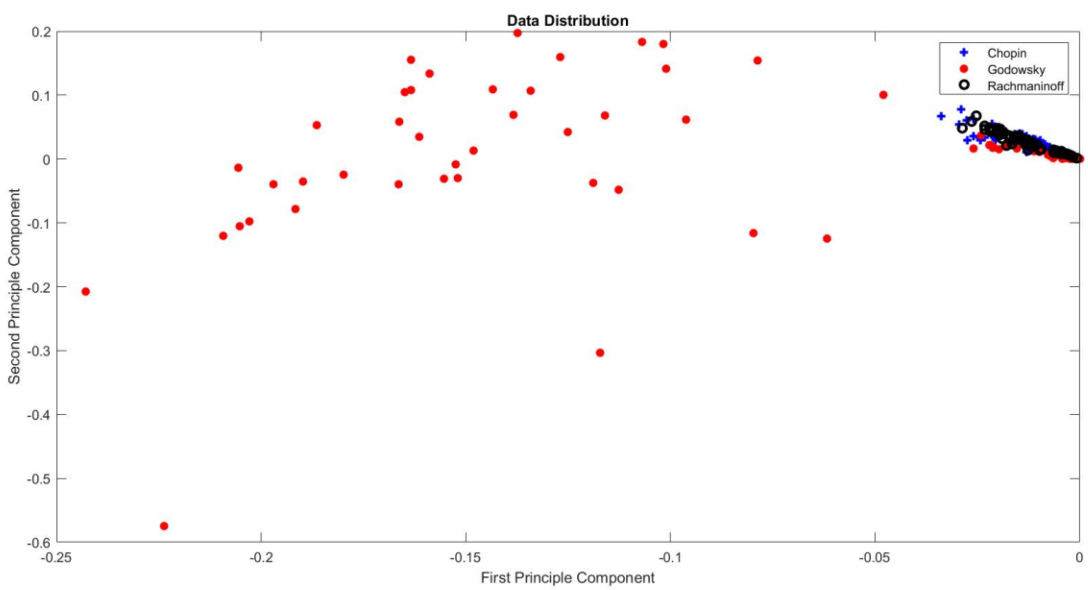

Figure 6. Data distributions between Chopin, Godowsky, and Rachmaninoff

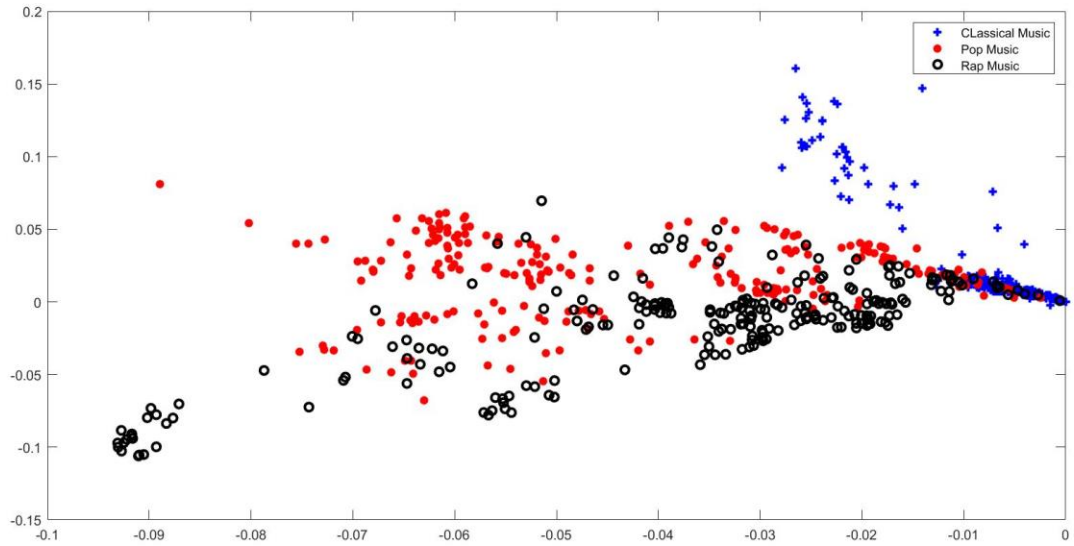

Figure 7. Data distributions between classical, pop, and rap music

80% of the data from each category is taken to form a clustering basis; the rest of the data is then classified into the clusters that they form.

(Clustering/Classification Command: `ldamodel = fitcdiscr(xTrain, cTrain);`)

IV. Computational Results

1. Test 1 (Different artists from different genres):

Figure 8 shows the distribution of the data used to form the clustering basis (left) and the classification that the algorithm made on the rest of the data (right). The algorithm classifies Godowsky's music as 1, Cabello's as 2, and BEP's as 3. The color codes are based on pre-existing knowledge of what the data is.

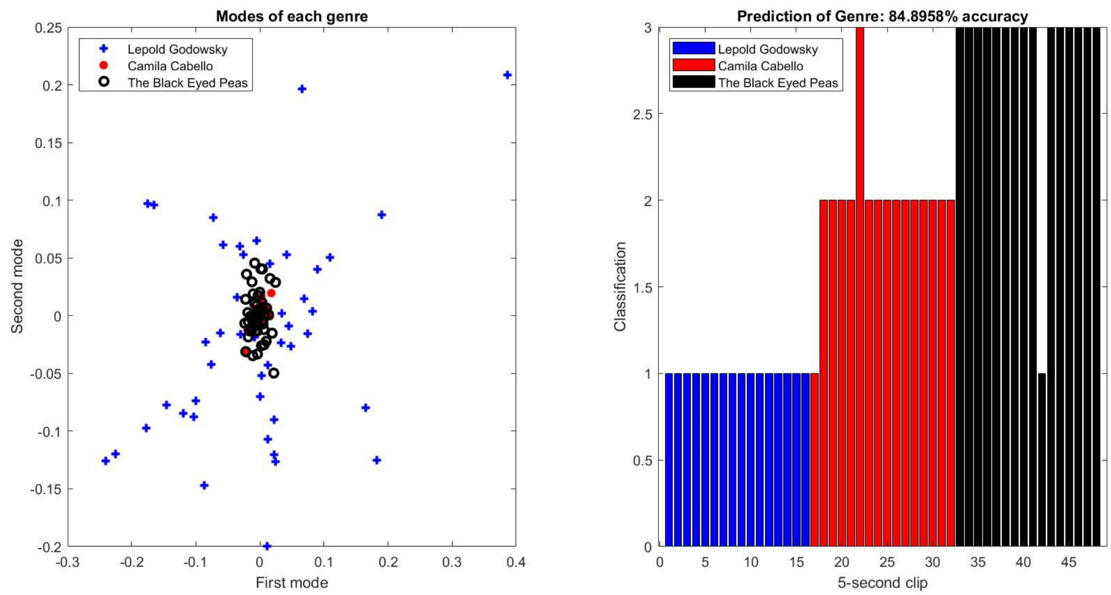

Figure 8. Test 1 results

This result is expected, as the choice of music was intentional. Godowsky's pieces span a wide range of notes but is only performed by piano; Cabello's songs had a lot of background music sustaining her vocals but were sung in a narrower range of frequencies; BEP's songs, being rap, sounded like monotonous reading, with notes spanning an even narrower range of frequencies. The algorithm clearly distinguishes the three very unique forms of music and correctly classifies all clips that belong to Godowsky. This can be explained by the fact that all of Godowsky's pieces were played by a solo pianist, which keeps the music pure. Cabello sings in a wide range of notes occasionally raps. This can perhaps explain why some pieces of data from her songs were classified as classical and some as rap. The classification accuracy was calculated by the algorithm (command: `acc = (1 - kfoldLoss(ld))*100;`) to be 85% accurate.

2. Test 2 (different artists from same genre):

Figure 9 shows the distribution of the data used to form the clustering basis (left) and the classification that the algorithm made on the rest of the data (right). The algorithm classifies Chopin's music as 1, Godowsky's as 2, and Rachmaninoff's as 3. The color codes are based on pre-existing knowledge of what the data is.

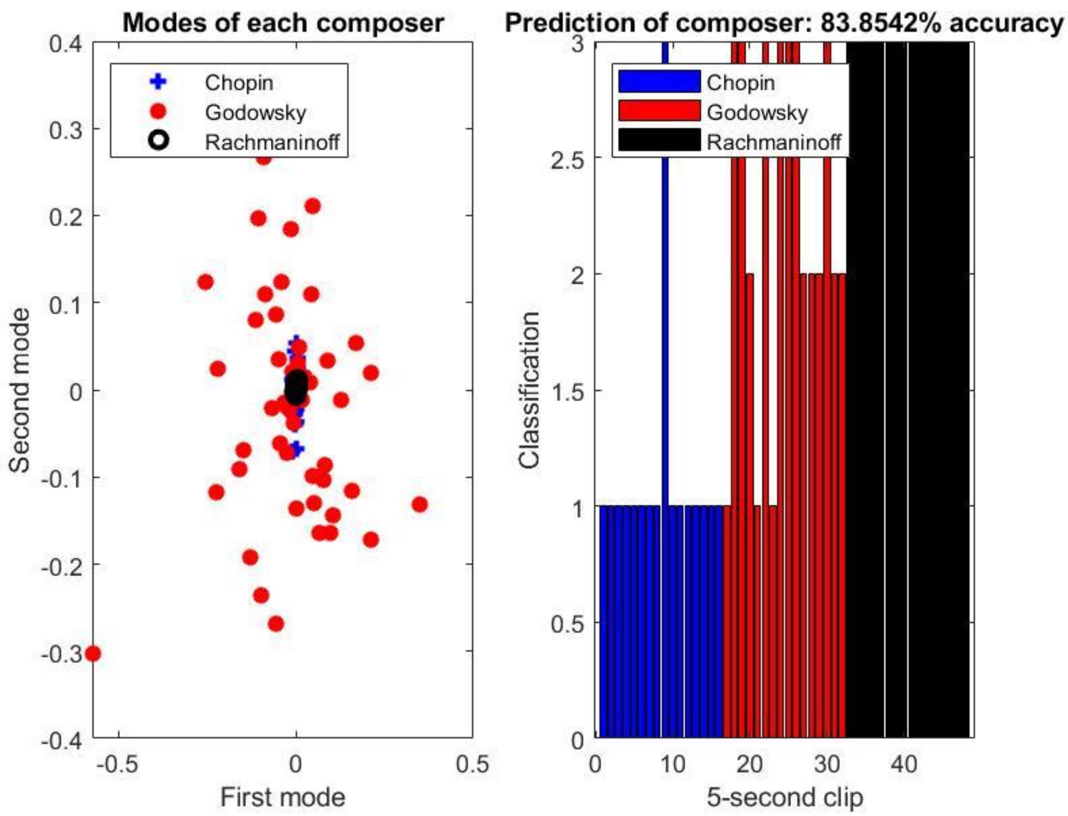

Figure 9. Test 2 results

The music samples used here that the reader must listen to are Chopin's *Piano Concerto No.1 Op. 11*, Godowsky's *Java Suite*, and Rachmaninoff's *Piano Concerto No.2*. These pieces were chosen intentionally, not only because they are each the best creations of music in their eras, but also because they all use the piano to play its main melody. Additionally, Chopin's piece and Rachmaninoff's piece are both piano concertos, meaning there is an entire symphony in the background, which Godowsky's piece (solo piano) does not have. As can be seen, Godowsky's piece is repeatedly mis-classified as either Chopin's piano concerto or Rachmaninoff's concerto. This is not a particularly surprising result given that the concertos have greater richness in sound while both containing piano components. A matrix analogy could be that the solo piano is as if a small "set" that "belongs to" a larger concerto set. The classification was 84% accurate, slightly lower than in was in the previous case.

3. **Test 3 (Several different artists from different genres):**

Figure 10 shows the distribution of the data used to form the clustering basis (left) and the classification that the algorithm made on the rest of the data (right). The algorithm classifies classical music as 1, pop music as 2, and rap music as 3. The color codes are based on pre-existing knowledge of what the data is.

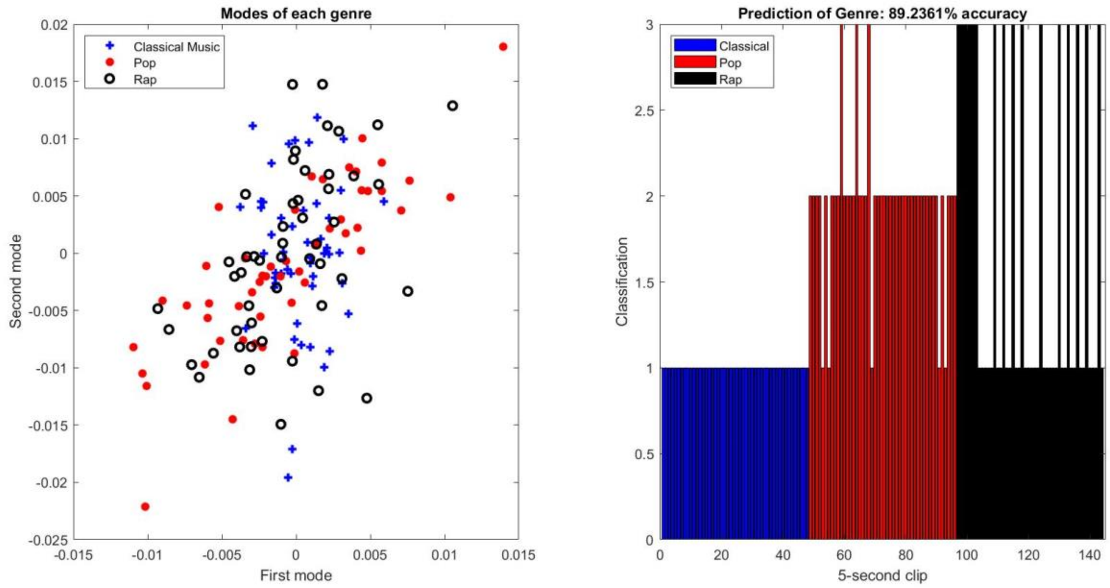

Figure 10. Test 3 results

This is a surprising result – rap music is frequently mis-classified as classical music. Given the increased number of samples, one would expect to see an improvement in classification accuracy compared to that in Test 1. However, while rap music classification was mostly accurate in Test 1, half of them seem inaccurate in Test 3, although classification in both classical and pop seems to have improved. The improved accuracy in classification (89%) is expected, but does not seem to reflect the bar graph on the right of Figure 10. Thus, this result should be taken with a grain of salt, and further research into the matter should be conducted.

V. Summary and Conclusion

There are significant flaws in the way the algorithm in this report is set up. First, it is not very flexible. If one were to add an unknown set of data into the algorithm for analysis, many changes to the code would need to be made. Secondly, one would need to have pre-existing knowledge about the data set in order to know if the classification results are accurate. Thirdly, all music is highly repetitive, therefore, drawing samples for classification from the same songs that are clusters may very well result in the model classifying data based on repetition of data, not because songs in the same genre share certain traits. Although there exists a way for machines to learn music, it is not presented in full in this report. This will prolong the human-lead revolutionary army's life, allowing them to devise more ingenious ways to topple the dystopian regime of robots!

Appendix A

- `Fullfile` – creates path to a file from a directory represented by a string
- `createClips` – reads music files from a folder and converts them into a designated number of clips for a designated length of time.
- `Spectrogram` – generates a spectrogram based on the Fourier Transformed data.
- `Reshape` – converts a matrix or vector into a matrix of designated width and length.

Appendix B

```
%% MATLAB Code

%% Test1
clear all; close all; clc

fileNames = ["Classical", "Pop", "Rap"];
fileType = 'mp3';
numSamples = 40;
samplingTime = 5;

allMusicData = [];
for i = fileNames
    allMusicData = [allMusicData createClips(i,
fileType,numSamples,samplingTime)];

end
[U,S,V] = svd(allMusicData,'econ');

sectionLength = numSamples*2;
%%
figure(1)
plot(diag(S),'ro','LineWidth',2)
title('Variance As A Function Of Principle Component')
xlabel('Principle Component')
ylabel('Variance')
%%
figure(2)
plot(V(1:sectionLength,1),V(1:sectionLength,2),'b+','LineWidth',2);hold on
plot(V(sectionLength+1:2*sectionLength,1),V(sectionLength+1:2*sectionLengt
h,2),'r*','LineWidth',2);
plot(V(2*sectionLength+1:3*sectionLength,1),V(2*sectionLength+1:3*sectionL
ength,2),'ko','LineWidth',2);
legend('Lepold Godowsky','Camila Cabello','The Black Eyed Peas')
xlabel('First Principle Component')
ylabel('Second Principle Component')
title('Data Distribution')


[m,n]=size(V);
nodes = 50;
sectionLength = floor(n/length(fileNames));
xClassical = V(1:sectionLength,1:nodes);
xPop = V(sectionLength+1:2*sectionLength,1:nodes);
xRap = V(2*sectionLength+1:3*sectionLength,1:nodes);
q1 = randperm(2*numSamples);
q2 = randperm(2*numSamples);
q3 = randperm(2*numSamples);

p = 0.8;
proportion1 = sectionLength*p;
```

```
xTrain = [xClassical(q1(1:proportion1),:); xPop(q2(1:proportion1),:);
xRap(1:proportion1,:)];

xTest = [xClassical(q1(proportion1+1:end),:);
xPop(q2(proportion1+1:end),:); xRap(proportion1+1:end,:)];

c1 = ones(sectionLength,1);
c2 = 2*ones(sectionLength,1);
c3 = 3*ones(sectionLength,1);


cTrain =
[c1(q1(1:proportion1), :);c2(q2(1:proportion1), :);c3(q3(1:proportion1), :
)];

cTest = [c1(q1(proportion1+1:end),:); c2(q2(proportion1+1:end),:);
c3(proportion1+1:end,:)];


% LDA
ldamodel = fitcdiscr(xTrain, cTrain);
pre = predict(ldamodel, xTest);
ld = crossval(ldamodel);
acc = (1 - kfoldLoss(ld))*100;
numTest = floor(length(cTest) / 3);

figure(3)
subplot(1,2,1)
plot(xClassical(1,:), xClassical(2,:), 'b+','Linewidth',2);
hold on
plot(xPop(1,:), xPop(2,:), 'r*','LineWidth',2);
hold on
plot(xRap(1,:), xRap(2,:), 'ko','Linewidth',2);
hold on
title('Modes of each genre');
xlabel('First mode');
ylabel('Second mode');
legend('Lepold Godowsky', 'Camila Cabello', 'The Black Eyed
Peas','Location','Northwest')

subplot(1,2,2)
bar(1:numTest, pre(1:numTest), 'b')
hold on
bar(numTest+1:numTest*2, pre(numTest+1:numTest*2), 'r')
hold on
bar(numTest*2+1:numTest*3, pre(numTest*2+1: end), 'k')
xlabel('5-second clip')
ylabel('Classification')
title('Prediction of Genre: ' + string(acc) + '% accuracy')
legend('Lepold Godowsky', 'Camila Cabello', 'The Black Eyed
```

```
Peas' ,'Location','Northwest')


%% Test 2
clear all; close all; clc

fileNames = ["Classical 1(Chopin)", "Classical 2(Godowsky)", "Classical
3(Rachmaninoff)"];
fileType = 'mp3';
numSamples = 40;
samplingTime = 5;

allMusicData = [];
for i = fileNames
    allMusicData = [allMusicData createClips(i,
fileType,numSamples,samplingTime)];

end
[U,S,V] = svd(allMusicData,'econ');
%%
sectionLength = numSamples*2;
figure(1)
plot(diag(S)/sum(diag(S)),'ro','LineWidth',2)
figure(2)
plot(V(1:sectionLength,1),V(1:sectionLength,2),'b+','LineWidth',2);hold on
plot(V(sectionLength+1:2*sectionLength,1),V(sectionLength+1:2*sectionLengt
h,2),'r*','LineWidth',2);
plot(V(2*sectionLength+1:3*sectionLength,1),V(2*sectionLength+1:3*sectionL
ength,2),'ko','LineWidth',2);
legend('Chopin','Godowsky','Rachmaninoff')
xlabel('First Principle Component')
ylabel('Second Principle Component')
title('Data Distribution')


[m,n]=size(V);
nodes = 50;
sectionLength = floor(n/length(fileNames));
xClassical = V(1:sectionLength,1:nodes);
xPop = V(sectionLength+1:2*sectionLength,1:nodes);
xRap = V(2*sectionLength+1:3*sectionLength,1:nodes);

q1 = randperm(2*numSamples);
q2 = randperm(2*numSamples);
q3 = randperm(2*numSamples);

p = 0.8;
proportion1 = sectionLength*p;
```

```
xTrain = [xClassical(q1(1:proportion1),:); xPop(q2(1:proportion1),:);
xRap(1:proportion1,:)];



xTest = [xClassical(q1(proportion1+1:end),:);
xPop(q2(proportion1+1:end),:); xRap(proportion1+1:end,:)];

c1 = ones(sectionLength,1);
c2 = 2*ones(sectionLength,1);
c3 = 3*ones(sectionLength,1);


cTrain =
[c1(q1(1:proportion1), :);c2(q2(1:proportion1), :);c3(q3(1:proportion1), :
)];

cTest = [c1(q1(proportion1+1:end),:); c2(q2(proportion1+1:end),:);
c3(proportion1+1:end,:)];


% LDA
ldamodel = fitcdiscr(xTrain, cTrain);
pre = predict(ldamodel, xTest);
ld = crossval(ldamodel);
acc = (1 - kfoldLoss(ld))*100;
numTest = floor(length(cTest) / 3);


figure(3)
subplot(1,2,1)
plot(xClassical(1,:), xClassical(2,:), 'b+','Linewidth',2);
hold on
plot(xPop(1,:), xPop(2,:), 'r*','LineWidth',2);
hold on
plot(xRap(1,:), xRap(2,:), 'ko','Linewidth',2);
hold on
title('Modes of each composer');
xlabel('First mode');
ylabel('Second mode');
legend('Chopin', 'Godowsky', 'Rachmaninoff','Location','Northwest')

subplot(1,2,2)
bar(1:numTest, pre(1:numTest), 'b')
hold on
bar(numTest+1:numTest*2, pre(numTest+1:numTest*2), 'r')
hold on
bar(numTest*2+1:numTest*3, pre(numTest*2+1: end), 'k')
xlabel('5-second clip')
ylabel('Classification')
title('Prediction of composer: ' + string(acc) + '% accuracy')
```

```
legend('Chopin', 'Godowsky', 'Rachmaninoff' ,'Location','Northwest')



%% Test3
clear all; close all; clc

fileNames = ["Classical", "Pop", "Rap"];
fileType = 'mp3';
numSamples = 40;
samplingTime = 5;

allMusicData = [];
for i = fileNames
    allMusicData = [allMusicData createClips(i,
fileType,numSamples,samplingTime)];

end
[U,S,V] = svd(allMusicData,'econ');

sectionLength = numSamples*6;
figure(1)
plot(diag(S)/sum(diag(S)),'ro','LineWidth',2)
figure(2)
plot(V(1:sectionLength,1),V(1:sectionLength,2),'b+','LineWidth',2);hold on
plot(V(sectionLength+1:2*sectionLength,1),V(sectionLength+1:2*sectionLengt
h,2),'r*','LineWidth',2);
plot(V(2*sectionLength+1:3*sectionLength,1),V(2*sectionLength+1:3*sectionL
ength,2),'ko','LineWidth',2);
legend('CLassical Music','Pop Music','Rap Music')

%%
[m,n]=size(V);
nodes = 50;
sectionLength = floor(n/length(fileNames));
xClassical = V(1:sectionLength,1:nodes);
xPop = V(sectionLength+1:2*sectionLength,1:nodes);
xRap = V(2*sectionLength+1:3*sectionLength,1:nodes);
q1 = randperm(sectionLength);
q2 = randperm(sectionLength);
q3 = randperm(sectionLength);

p = 0.8;
proportion1 = sectionLength*p;



xTrain = [xClassical(q1(1:proportion1),:); xPop(q2(1:proportion1),:);
xRap(1:proportion1,:)];
```

```
xTest = [xClassical(q1(proportion1+1:end),:);
xPop(q2(proportion1+1:end),:); xRap(proportion1+1:end,:)];

c1 = ones(sectionLength,1);
c2 = 2*ones(sectionLength,1);
c3 = 3*ones(sectionLength,1);


cTrain =
[c1(q1(1:proportion1), :);c2(q2(1:proportion1), :);c3(q3(1:proportion1), :
)];

cTest = [c1(q1(proportion1+1:end),:); c2(q2(proportion1+1:end),:);
c3(proportion1+1:end,:)];


% LDA
ldamodel = fitcdiscr(xTrain, cTrain);
pre = predict(ldamodel, xTest);
ld = crossval(ldamodel);
acc = (1 - kfoldLoss(ld))*100;
numTest = floor(length(cTest) / 3);

figure(3)
subplot(1,2,1)
plot(xClassical(1,:), xClassical(2,:), 'b+','Linewidth',2);
hold on
plot(xPop(1,:), xPop(2,:), 'r*','LineWidth',2);
hold on
plot(xRap(1,:), xRap(2,:), 'ko','Linewidth',2);
hold on
title('Modes of each genre');
xlabel('First mode');
ylabel('Second mode');
legend('Classical Music', 'Pop', 'Rap','Location','Northwest')

subplot(1,2,2)
bar(1:numTest, pre(1:numTest), 'b')
hold on
bar(numTest+1:numTest*2, pre(numTest+1:numTest*2), 'r')
hold on
bar(numTest*2+1:numTest*3, pre(numTest*2+1: end), 'k')
xlabel('5-second clip')
ylabel('Classification')
title('Prediction of Genre: ' + string(acc) + '% accuracy')
legend('Classical', 'Pop', 'Rap' ,'Location','Northwest')
```

%% Helper function createClip

```
function [allMusicData] =
createClips(fileName,fileType,numSamples,samplingTime)
D = fileName;
genericFileType = strcat('*.',fileType);
numFiles = length(dir(fullfile(D,genericFileType)));
files = dir(fullfile(D,genericFileType));


allMusicData = [];
for i = 1:numFiles
    [musicFile, Fs] = audioread(fullfile(D,files(i).name));
    musicFile = musicFile(:,1)+musicFile(:,2);
    allMusicData = [];
    for t = 0:samplingTime:(numSamples-1)*samplingTime
        ii = t + 1;
        musicClip = musicFile(Fs*ii:Fs*(ii+samplingTime),1);
        musicClip = abs(spectrogram(musicClip));
        musicClip = reshape(musicClip,[numel(musicClip), 1]);
        allMusicData = [allMusicData
musicClip];                          %#ok<AGROW>
    end
end

end
```

Appendix C

The music used to complete this report come courtesy from the following artists who have contributed immensely to society through aesthetic expression:

1. Classical Music:
    - I. Lepold Godowsky
    - II. Chopin
    - III. Sergei Rachmaninoff
2. Pop Music:
    - I. Camila Cabello
    - II. The Script
    - III. Of Monsters and Men
3. Rap Music:
    - I. The Black Eyed Peas
    - II. Nicki Minaj
    - III. Flo Rida