UNIVERSITY OF WASHINGTON

AMATH 482 A WI 20: COMPUTATIONAL METHODS FOR DATA ANALYSIS

Homework 4

Abstract

The goal of this project is to analyze and sort different audio signals with machine learning.
We take music from different genres and analyze their frequencies. We take different properties of the frequency function and try to find similarities for the same genres. Now we can take those similarities and use them as data for training our program to look for these properties in new test data to classify their genres.

Ather Ahmad

Instructor: Craig GIN

1 Introduction and Overview

To deconstruct a signal we can use a Fourier transform and look at the frequencies produced by such. These frequencies can be compare and used to make differences between different genres of music. The properties obtained from the frequencies can be used to find clusters in a multidimensional function. To analyze ans classify sections we can use Principal component analysis (PCA) and Linear Discriminant Analysis (LDA).

2 Theoretical Background

2.1 Fourier Transform

The general idea behind the Fourier Transform is to obtain a spectrum of a given function. The most common use is to analyze a signal over time and transform it into the frequency space to find all frequencies which build that signal.

2.1.1 Continuous Fourier Transform

Definition of a Fourier Transform for a given function

$$\hat{f}(k) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} f(x)\; e^{-ikx} dx$$

and can be inverted back with

$$f(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} \hat{f}(k)\; e^{ikx} dk\ .$$

Instead of transforming the given function it can be decomposed in *sin* and *cos* terms with **Fourier coefficients (Fourier series)**

$$f(x) = \frac{a_0}{2} \sum_{k=1}^{\infty} \left[a_2\; cos(kx) + b_k\; sin(kx)\right]$$

with

$$a_k = \frac{1}{\pi} \int_{-\pi}^{\pi} f(x)\; cos(kx)\; dx \quad k \geq 0$$
$$b_k = \frac{1}{\pi} \int_{-\pi}^{\pi} f(x)\; sin(kx)\; dx \quad k > 0\ .$$

2.1.2 Discrete Fourier Transform

To calculate our function we have to discretize our space:

$$x \in \mathbb{R} \to x \in \{x_0,\ x_1,\ x_2,\ ...,\ x_{N-1}\}$$

this gives us new terms for the Fourier transform

$$\hat{x}_k = \sum_{n=0}^{N-1} x_n e^{-\frac{2\pi ikn}{N}}$$
$$x_k = \sum_{n=0}^{N-1} \hat{x}_k e^{\frac{2\pi ikn}{N}}$$

2.2 Principal component analysis (PCA)

The PCA helps us to structure, simplify and illustrate extensive data sets by approximating a large number of statistical variables by using a smaller number of meaningful linear combinations (the "main components"). The underlying data set typically has the structure of a matrix: p characteristics were measured on n subjects or objects. Such a data set can be illustrated as a set of n points in p-dimensional space $\mathbb{R}^p$. The PCA aims to project these data points into a q-dimensional subspace $\mathbb{R}^q$ $(q < p)$ in such a way that as little information as possible is lost and existing redundancy is summarized in the form of correlation in the data points.

A main axis transformation is carried out mathematically: The correlation of multidimensional features is minimized by transferring them to a vector space with a new basis.The main axis transformation can be specified by an orthogonal matrix, which is formed from the eigenvectors of the covariance matrix. The rotation of the coordinate system is carried out in such a way that the covariance matrix is diagonalized.

2.3 Linear Discriminant Analysis (LDA)

LDA is a method that assigns a score to each observation in the discriminant analysis. The group membership of each observation and the boundaries between the groups are determined from the score. If the observations are known to belong to a group, the feature variables are combined into a single discriminant variable with minimal loss of information.

Given N d-dimensional feature vectors x, of which n_1 belong to class C_1 and n_2 belong to class C_2. A discriminant function now describes the equation of a hyperplane that optimally separates the classes. Depending on the separability of the classes, there are linear and non-linear.

2.4 Classification

The mentioned methods were used to analyze the given train data and chose barriers for the given music genres (Hip Hop, Jazz and Electro) in a 3D space. With these given barriers a set of test data was used to find out if the genres are classified correctly.

3 Algorithm Implementation and Development

At first the songs are loaded into the file. Multiple samples from different songs are used as training data set. These adoi signals are separated into different properties to calculate their Fourier transform. The Frequencies are analyzed and the best options for properties are picked to distinguish different music genres. In this case we take a look at the average height of the intensity of the frequencies, the average absolute frequency value of the function and the number of frequency values which hit the $\frac{1}{1000}$ mark of the maximum intensity. These properties are used as the three dimensional variables in the 3D space. These 'clusters' of data points are portrayed together with the lines along the greatest variance, calculated with PCA methods. With these information 'barriers' are chosen to build sections in the 3D space to separate the different genres. With a new set of data with known genre these sections are tested for their accuracy.

4 Computational Results

The trained data already showed qualitatively good clusters. These data points are portrayed in Figure 1.

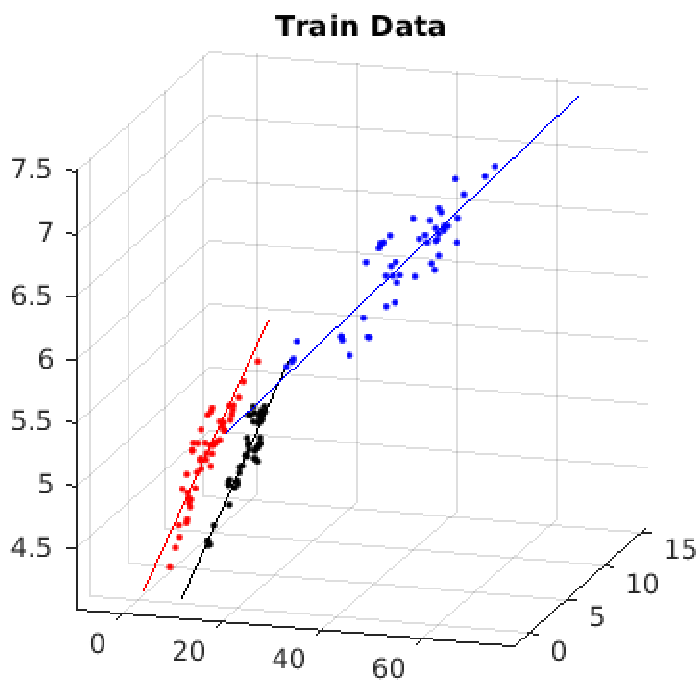

Figure 1: This shows the data sets of the training data (blue for Hip Hop, red for Jazz and black for Electro). The points are the data points for the different properties. The lines represent the direction of the highest variance.

The test data showed good results for two music genres:

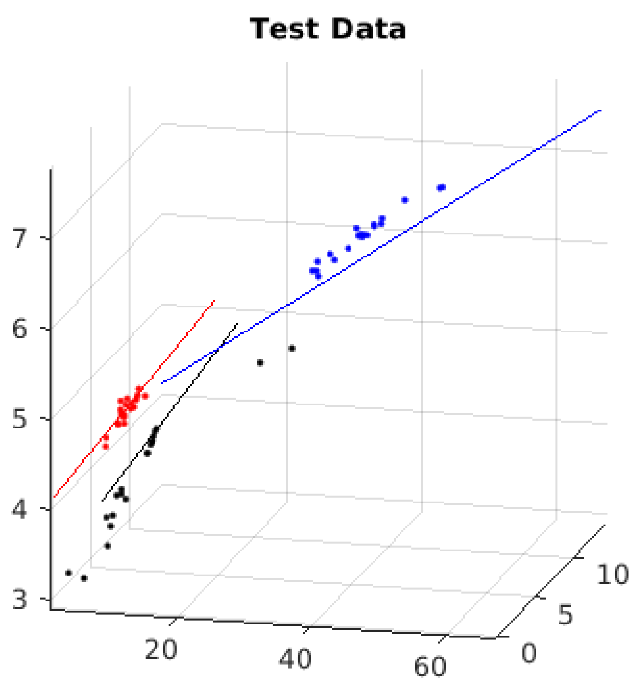

Figure 2: This shows the data sets of the tested data (blue for Hip Hop, red for Jazz and black for Electro). The points are the data points for the different properties. The lines represent the direction of the highest variance of the training data. Hip Hop and Jazz show good results since the data points are close to the trained lines. Electro has a high spread of points, which we didn't had before. This indicates some errors the code might make.

5 Summary and Conclusions

The accuracy for Hip Hop and Jazz were 100% for Electro it was 60%

6 Appendix A: MATLAB functions used and brief implementation explanation

6.1 Cell Array

Cell Array is a data type which can be indexed with k, where k lets us access any kind of data (a number, another array or a matrix e.g.) and use it in a loop.

6.2 Scatter3

Scatter3 lets us display dots in a 3D space with given x, y and z

6.3 Function: point_to_line_distance

This function was used to calculate the barriers. The used 'distance' was a mixture of distance (given by this function) between data point and the line along the direction of the highest variance of the training data and the distance between the mean of the training data and data point.

7 Appendix B: MATLAB codes

clear all, clc, close all

samples$_{p}er_{s}ong = 10$;

number$_{o}f_{s}ongs = 5$;

number$_{o}f_{s}amples = samples_{p}er_{s}ong * number_{o}f_{s}ongs$;

sample = cell(samples$_{p}er_{s}ong, 1)$;

for k = 1:samples$_{p}er_{s}ong$

samplek = [1500000+((k-1)*225000) 1500000+(k*225000)];

end

y$_{o}tis = cell(number_{o}f_{s}amples); Fs_{o}tis = cell(number_{o}f_{s}amples)$;

for k = 1:samples$_{p}er_{s}ong$

[y$_{o}tisk, Fs_{o}tisk] = audioread(''Behind_{C}losed_{D}oors(otis).mp3'', samplek)$;

[y$_{o}tisk + 1 * samples_{p}er_{s}ong, Fs_{o}tisk + 1 * samples_{p}er_{s}ong] = audioread(''La_{L}a_{L}a(otis).mp3'', samp$

[y$_{o}tisk + 2 * samples_{p}er_{s}ong, Fs_{o}tisk + 2 * samples_{p}er_{s}ong] = audioread(''Mighty_{F}ine(otis).mp3'', s$

[y$_{o}tisk + 3 * samples_{p}er_{s}ong, Fs_{o}tisk + 3 * samples_{p}er_{s}ong] = audioread(''Otis_{M}cMusic(otis).mp3''$

[y$_{o}tisk + 4 * samples_{p}er_{s}ong, Fs_{o}tisk + 4 * samples_{p}er_{s}ong] = audioread(''Sneaking_{o}n_{S}eptember(oti$
end
signal$_{o}tis = cell(number_{o}f_{s}amples, 1);$
for k = 1:number$_{o}f_{s}amples$
signal$_{o}tisk = y_{o}tisk(:, 1)';$
end
data$_{p}oints_{o}tis = cell(number_{o}f_{s}amples, 1); time_{o}tis = cell(number_{o}f_{s}amples, 1);$
for k = 1:number$_{o}f_{s}amples$
data$_{p}oints_{o}tisk = length(signal_{o}tisk); time_{o}tisk = data_{p}oints_{o}tisk/Fs_{o}tisk;$
end
data$_{v}ector_{o}tis = cell(number_{o}f_{s}amples, 1); time_{v}ector_{o}tis = cell(number_{o}f_{s}amples, 1);$
for k = 1:number$_{o}f_{s}amples$
data$_{v}ector_{o}tisk = 1 : data_{p}oints_{o}tisk; time_{v}ector_{o}tisk = data_{v}ector_{o}tisk/Fs_{o}tisk;$
end
figure(1)
for k = 1:number$_{o}f_{s}amples$
subplot(number$_{o}f_{s}amples, 1, k)plot(time_{v}ector_{o}tisk, signal_{o}tisk); axisoff$
end
fft$_{o}tis = cell(number_{o}f_{s}amples, 1); frequencies_{s}pace_{o}tis = cell(number_{o}f_{s}amples, 1); frequencies_{s}pa$
$cell(number_{o}f_{s}amples, 1);$
for k = 1:number$_{o}f_{s}amples$
fft$_{o}tisk = fft(signal_{o}tisk);$
frequencies$_{s}pace_{o}tisk = (2 * pi/time_{o}tisk * [0 : (data_{p}oints_{o}tisk/2) - data_{p}oints_{o}tisk/2 :$
$-1]); frequencies_{s}pace_{s}hifted_{o}tisk = fftshift(frequencies_{s}pace_{o}tisk);$
end
fourier$_{o}tis = cell(number_{o}f_{s}amples, 1);$
spread$_{f}actor = 1000;$
spread$_{v}ariable = 5;$
for k = 1:number$_{o}f_{s}amples$
fourier$_{o}tisk(1, :) = frequencies_{s}pace_{s}hifted_{o}tisk; fourier_{o}tisk(2, :) = fftshift(abs(fft_{o}tisk));$
end
length$_{f}ourier_{o}tis = cell(number_{o}f_{s}amples, 1);$
sum$_{f}ourier_{o}tis = cell(number_{o}f_{s}amples, 1);$
max$_{f}ourier_{o}tis = cell(number_{o}f_{s}amples, 1);$
mean$_{f}ourier_{o}tis = cell(number_{o}f_{s}amples, 1);$
spread$_{f}ourier_{o}tis = cell(number_{o}f_{s}amples, 1);$
for j = 1 : number$_{o}f_{s}amples$
length$_{f}ourier_{o}tisj = length(fourier_{o}tisj);$
max$_{f}ourier_{o}tisj = max(abs(fourier_{o}tisj(1, j)));$
sum$_{f}ourier_{o}tisj = 0;$
mean$_{f}ourier_{o}tisj = 0;$

spread$_{f}ourier_{o}tisj = 0$;

for k = 1 : length$_{f}ourier_{o}tisj$

sum$_{f}ourier_{o}tisj = sum_{f}ourier_{o}tisj + fourier_{o}tisj(2,k);$

mean$_{f}ourier_{o}tisj = mean_{f}ourier_{o}tisj + abs(fourier_{o}tisj(1,k)) * fourier_{o}tisj(2,k)/max_{f}ourier_{o}tisj;$

if fourier$_{o}tisj(2,k) > (max_{f}ourier_{o}tisj/spread_{f}actor)$

spread$_{f}ourier_{o}tisj = spread_{f}ourier_{o}tisj + 1;$

end

end

sum$_{f}ourier_{o}tisj = sum_{f}ourier_{o}tisj/length_{f}ourier_{o}tisj;$

mean$_{f}ourier_{o}tisj = mean_{f}ourier_{o}tisj/length_{f}ourier_{o}tisj;$

spread$_{f}ourier_{o}tisj = (spread_{f}ourier_{o}tisj)^{(}1/spread_{v}ariable);$

end

y$_{j}ammy = cell(number_{o}f_{s}amples); Fs_{j}ammy = cell(number_{o}f_{s}amples);$

for k = 1:samples$_{p}er_{s}ong$

[y$_{j}ammyk, Fs_{j}ammyk] = audioread(''Happy_{B}irthday(jammy).mp3'', samplek);$

[y$_{j}ammyk + 1 * samples_{p}er_{s}ong, Fs_{j}ammyk + 1 * samples_{p}er_{s}ong] = audioread(''Maple_{L}eaf_{R}ag(jan$

[y$_{j}ammyk + 2 * samples_{p}er_{s}ong, Fs_{j}ammyk + 2 * samples_{p}er_{s}ong] = audioread(''Minor_{B}lues_{f}or_{B}oo$

[y$_{j}ammyk + 3 * samples_{p}er_{s}ong, Fs_{j}ammyk + 3 * samples_{p}er_{s}ong] = audioread(''Present_{D}ay(jamm$

[y$_{j}ammyk + 4 * samples_{p}er_{s}ong, Fs_{j}ammyk + 4 * samples_{p}er_{s}ong] = audioread(''When_{J}ohnny_{G}oes_{N}$

end

signal$_{j}ammy = cell(number_{o}f_{s}amples, 1);$

for k = 1:number$_{o}f_{s}amples$

signal$_{j}ammyk = y_{j}ammyk(:,1)';$

end

data$_{p}oints_{j}ammy = cell(number_{o}f_{s}amples, 1); time_{j}ammy = cell(number_{o}f_{s}amples, 1);$

for k = 1:number$_{o}f_{s}amples$

data$_{p}oints_{j}ammyk = length(signal_{j}ammyk); time_{j}ammyk = data_{p}oints_{j}ammyk/Fs_{j}ammyk;$

end

data$_{v}ector_{j}ammy = cell(number_{o}f_{s}amples, 1); time_{v}ector_{j}ammy = cell(number_{o}f_{s}amples, 1);$

for k = 1:number$_{o}f_{s}amples$

data$_{v}ector_{j}ammyk = 1 : data_{p}oints_{j}ammyk; time_{v}ector_{j}ammyk = data_{v}ector_{j}ammyk/Fs_{j}ammyk;$

end

figure(2)

for k = 1:number$_{o}f_{s}amples$

subplot(number$_{o}f_{s}amples, 1, k)plot(time_{v}ector_{j}ammyk, signal_{j}ammyk); axisoff$

end

fft$_{j}ammy = cell(number_{o}f_{s}amples, 1); frequencies_{s}pace_{j}ammy = cell(number_{o}f_{s}amples, 1); frequen$
$cell(number_{o}f_{s}amples, 1);$

for k = 1:number$_{o}f_{s}amples$

fft$_{j}ammyk = fft(signal_{j}ammyk);$

frequencies$_{s}pace_{j}ammyk = (2 * pi/time_{j}ammyk * [0 : (data_{p}oints_{j}ammyk/2) - data_{p}oints_{j}ammyk/2 : -1]); frequencies_{s}pace_{s}hifted_{j}ammyk = fftshift(frequencies_{s}pace_{j}ammyk);$

end

fourier$_{j}ammy = cell(number_{o}f_{s}amples, 1);$

for k = 1:number$_{o}f_{s}amples$

fourier$_{j}ammyk(1, :) = frequencies_{s}pace_{s}hifted_{j}ammyk; fourier_{j}ammyk(2, :) = fftshift(abs(fft_{j}am$

end

length$_{f}ourier_{j}ammy = cell(number_{o}f_{s}amples, 1);$

sum$_{f}ourier_{j}ammy = cell(number_{o}f_{s}amples, 1);$

max$_{f}ourier_{j}ammy = cell(number_{o}f_{s}amples, 1);$

mean$_{f}ourier_{j}ammy = cell(number_{o}f_{s}amples, 1);$

spread$_{f}ourier_{j}ammy = cell(number_{o}f_{s}amples, 1);$

for j = 1 : number$_{o}f_{s}amples$

length$_{f}ourier_{j}ammyj = length(fourier_{j}ammyj);$

max$_{f}ourier_{j}ammyj = max(abs(fourier_{j}ammyj(1, j)));$

sum$_{f}ourier_{j}ammyj = 0;$

mean$_{f}ourier_{j}ammyj = 0;$

spread$_{f}ourier_{j}ammyj = 0;$

for k = 1 : length$_{f}ourier_{j}ammyj$

sum$_{f}ourier_{j}ammyj = sum_{f}ourier_{j}ammyj + fourier_{j}ammyj(2, k);$

mean$_{f}ourier_{j}ammyj = mean_{f}ourier_{j}ammyj + abs(fourier_{j}ammyj(1, k)) * fourier_{j}ammyj(2, k)/max$

if fourier$_{j}ammyj(2, k) > (max_{f}ourier_{j}ammyj/spread_{f}actor)$

spread$_{f}ourier_{j}ammyj = spread_{f}ourier_{j}ammyj + 1;$

end

end

sum$_{f}ourier_{j}ammyj = sum_{f}ourier_{j}ammyj/length_{f}ourier_{j}ammyj;$

mean$_{f}ourier_{j}ammyj = mean_{f}ourier_{j}ammyj/length_{f}ourier_{j}ammyj;$

spread$_{f}ourier_{j}ammyj = (spread_{f}ourier_{j}ammyj)^{(}1/spread_{v}ariable);$

end

y$_{m}ax = cell(number_{o}f_{s}amples); Fs_{m}ax = cell(number_{o}f_{s}amples);$

for k = 1:samples$_{p}er_{s}ong$

[y$_{m}axk, Fs_{m}axk] = audioread(''All_{I}Can_{D}o_{I}s_{T}his(max).mp3'', samplek);$

[y$_{m}axk + 1 * samples_{p}er_{s}ong, Fs_{m}axk + 1 * samples_{p}er_{s}ong] = audioread(''Come_{W}ith_{S}ome_{F}unk(ma$

[y$_{m}axk + 2 * samples_{p}er_{s}ong, Fs_{m}axk + 2 * samples_{p}er_{s}ong] = audioread(''Mom_{s}{}_{H}ouse(max).mp3''$

[y$_{m}axk + 3 * samples_{p}er_{s}ong, Fs_{m}axk + 3 * samples_{p}er_{s}ong] = audioread(''Sense_{o}f_{H}umor(max).mp$

[y$_{m}axk + 4 * samples_{p}er_{s}ong, Fs_{m}axk + 4 * samples_{p}er_{s}ong] = audioread(''Waterfront_{P}roperty(ma$

end

signal$_{m}ax = cell(number_{o}f_{s}amples, 1);$

for k = 1:number$_{o}f_{s}amples$

signal$_{m}axk = y_{m}axk(:, 1)';$

end

data$_{p}oints_{m}ax = cell(number_{o}f_{s}amples, 1); time_{m}ax = cell(number_{o}f_{s}amples, 1);$

for k = 1:number$_{o}f_{s}amples$

data$_{p}oints_{m}axk = length(signal_{m}axk); time_{m}axk = data_{p}oints_{m}axk/Fs_{m}axk;$

end

data$_{v}ector_{m}ax = cell(number_{o}f_{s}amples, 1); time_{v}ector_{m}ax = cell(number_{o}f_{s}amples, 1);$

for k = 1:number$_{o}f_{s}amples$

data$_{v}ector_{m}axk = 1 : data_{p}oints_{m}axk; time_{v}ector_{m}axk = data_{v}ector_{m}axk/Fs_{m}axk;$

end

figure(3)

for k = 1:number$_{o}f_{s}amples$

subplot(number$_{o}f_{s}amples, 1, k)plot(time_{v}ector_{m}axk, signal_{m}axk); axisoff$

end

fft$_{m}ax = cell(number_{o}f_{s}amples, 1); frequencies_{s}pace_{m}ax = cell(number_{o}f_{s}amples, 1); frequencies_{s}p$
$cell(number_{o}f_{s}amples, 1);$

for k = 1:number$_{o}f_{s}amples$

fft$_{m}axk = fft(signal_{m}axk);$

frequencies$_{s}pace_{m}axk = (2 * pi/time_{m}axk * [0 : (data_{p}oints_{m}axk/2) - data_{p}oints_{m}axk/2 :$
$-1]); frequencies_{s}pace_{s}hifted_{m}axk = fftshift(frequencies_{s}pace_{m}axk);$

end

fourier$_{m}ax = cell(number_{o}f_{s}amples, 1);$

for k = 1:number$_{o}f_{s}amples$

fourier$_{m}axk(1, :) = frequencies_{s}pace_{s}hifted_{m}axk; fourier_{m}axk(2, :) = fftshift(abs(fft_{m}axk));$

end

length$_{f}ourier_{m}ax = cell(number_{o}f_{s}amples, 1);$

sum$_{f}ourier_{m}ax = cell(number_{o}f_{s}amples, 1);$

max$_{f}ourier_{m}ax = cell(number_{o}f_{s}amples, 1);$

mean$_{f}ourier_{m}ax = cell(number_{o}f_{s}amples, 1);$

spread$_{f}ourier_{m}ax = cell(number_{o}f_{s}amples, 1);$

for j = 1 : number$_{o}f_{s}amples$

length$_{f}ourier_{m}axj = length(fourier_{m}axj);$

max$_{f}ourier_{m}axj = max(abs(fourier_{m}axj(1, j)));$

sum$_{f}ourier_{m}axj = 0;$

mean$_{f}ourier_{m}axj = 0;$

spread$_{f}ourier_{m}axj = 0;$

for k = 1 : length$_{f}ourier_{m}axj$

sum$_{f}ourier_{m}axj = sum_{f}ourier_{m}axj + fourier_{m}axj(2, k);$

mean$_{f}ourier_{m}axj = mean_{f}ourier_{m}axj + abs(fourier_{m}axj(1, k)) * fourier_{m}axj(2, k)/max_{f}ourier_{m}ax$

if fourier$_{m}axj(2, k) > (max_{f}ourier_{m}axj/spread_{f}actor)$

spread$_{f}ourier_{m}axj = spread_{f}ourier_{m}axj + 1;$

end

end

sum$_{f}ourier_{m}axj = sum_{f}ourier_{m}axj/length_{f}ourier_{m}axj;$
mean$_{f}ourier_{m}axj = mean_{f}ourier_{m}axj/length_{f}ourier_{m}axj;$
spread$_{f}ourier_{m}axj = (spread_{f}ourier_{m}axj)^{(}1/spread_{v}ariable);$
end
figure(4)
sum$_{f}ourier_{o}tis_{p}lot = zeros(1, number_{o}f_{s}amples);$
mean$_{f}ourier_{o}tis_{p}lot = zeros(1, number_{o}f_{s}amples);$
spread$_{f}ourier_{o}tis_{p}lot = zeros(1, number_{o}f_{s}amples);$
for k = 1:number$_{o}f_{s}amples$
sum$_{f}ourier_{o}tis_{p}lot(k) = sum_{f}ourier_{o}tisk;$
mean$_{f}ourier_{o}tis_{p}lot(k) = mean_{f}ourier_{o}tisk;$
spread$_{f}ourier_{o}tis_{p}lot(k) = spread_{f}ourier_{o}tisk;$
end
scatter3(sum$_{f}ourier_{o}tis_{p}lot, mean_{f}ourier_{o}tis_{p}lot, spread_{f}ourier_{o}tis_{p}lot,' blue')holdonaxisvis3d$
sum$_{f}ourier_{j}ammy_{p}lot = zeros(1, number_{o}f_{s}amples);$
mean$_{f}ourier_{j}ammy_{p}lot = zeros(1, number_{o}f_{s}amples);$
spread$_{f}ourier_{j}ammy_{p}lot = zeros(1, number_{o}f_{s}amples);$
for k = 1:number$_{o}f_{s}amples$
sum$_{f}ourier_{j}ammy_{p}lot(k) = sum_{f}ourier_{j}ammyk;$
mean$_{f}ourier_{j}ammy_{p}lot(k) = mean_{f}ourier_{j}ammyk;$
spread$_{f}ourier_{j}ammy_{p}lot(k) = spread_{f}ourier_{j}ammyk;$
end
scatter3(sum$_{f}ourier_{j}ammy_{p}lot, mean_{f}ourier_{j}ammy_{p}lot, spread_{f}ourier_{j}ammy_{p}lot,' red')$
sum$_{f}ourier_{m}ax_{p}lot = zeros(1, number_{o}f_{s}amples);$
mean$_{f}ourier_{m}ax_{p}lot = zeros(1, number_{o}f_{s}amples);$
spread$_{f}ourier_{m}ax_{p}lot = zeros(1, number_{o}f_{s}amples);$
for k = 1:number$_{o}f_{s}amples$
sum$_{f}ourier_{m}ax_{p}lot(k) = sum_{f}ourier_{m}axk;$
mean$_{f}ourier_{m}ax_{p}lot(k) = mean_{f}ourier_{m}axk;$
spread$_{f}ourier_{m}ax_{p}lot(k) = spread_{f}ourier_{m}axk;$
end
scatter3(sum$_{f}ourier_{m}ax_{p}lot, mean_{f}ourier_{m}ax_{p}lot, spread_{f}ourier_{m}ax_{p}lot,' black')$
mean$_{o}tis(1,1) = mean(sum_{f}ourier_{o}tis_{p}lot);$
mean$_{o}tis(1,2) = mean(mean_{f}ourier_{o}tis_{p}lot);$
mean$_{o}tis(1,3) = mean(spread_{f}ourier_{o}tis_{p}lot);$
scatter3(mean$_{o}tis(1,1), mean_{o}tis(1,2), mean_{o}tis(1,3),' yellow')$
mean$_{j}ammy(1,1) = mean(sum_{f}ourier_{j}ammy_{p}lot);$
mean$_{j}ammy(1,2) = mean(mean_{f}ourier_{j}ammy_{p}lot);$
mean$_{j}ammy(1,3) = mean(spread_{f}ourier_{j}ammy_{p}lot);$
scatter3(mean$_{j}ammy(1,1), mean_{j}ammy(1,2), mean_{j}ammy(1,3),' yellow')$
mean$_{m}ax(1,1) = mean(sum_{f}ourier_{m}ax_{p}lot);$

mean$_{m}ax(1,2) = mean(mean_{f}ourier_{m}ax_{p}lot);$
mean$_{m}ax(1,3) = mean(spread_{f}ourier_{m}ax_{p}lot);$
scatter3(mean$_{m}ax(1,1), mean_{m}ax(1,2), mean_{m}ax(1,3),' yellow')$
figure(5)
zero$_{m}ean_{o}tis(1,:) = sum_{f}ourier_{o}tis_{p}lot - mean_{o}tis(1,1);$
zero$_{m}ean_{o}tis(2,:) = mean_{f}ourier_{o}tis_{p}lot - mean_{o}tis(1,2);$
zero$_{m}ean_{o}tis(3,:) = spread_{f}ourier_{o}tis_{p}lot - mean_{o}tis(1,3);$
scatter3(zero$_{m}ean_{o}tis(1,:), zero_{m}ean_{o}tis(2,:), zero_{m}ean_{o}tis(3,:),' blue')axisvis3d$
covarience$_{s}um_{o}tis = cov(zero_{m}ean_{o}tis(1,:));$
covariance$_{m}ean_{o}tis = cov(zero_{m}ean_{o}tis(2,:));$
covariance$_{s}pread_{o}tis = cov(zero_{m}ean_{o}tis(3,:));$
figure(6)
zero$_{m}ean_{j}ammy(1,:) = sum_{f}ourier_{j}ammy_{p}lot - mean_{j}ammy(1,1);$
zero$_{m}ean_{j}ammy(2,:) = mean_{f}ourier_{j}ammy_{p}lot - mean_{j}ammy(1,2);$
zero$_{m}ean_{j}ammy(3,:) = spread_{f}ourier_{j}ammy_{p}lot - mean_{j}ammy(1,3);$
scatter3(zero$_{m}ean_{j}ammy(1,:), zero_{m}ean_{j}ammy(2,:), zero_{m}ean_{j}ammy(3,:),' red')axisvis3d$
covarience$_{s}um_{j}ammy = cov(zero_{m}ean_{j}ammy(1,:));$
covariance$_{m}ean_{j}ammy = cov(zero_{m}ean_{j}ammy(2,:));$
covariance$_{s}pread_{j}ammy = cov(zero_{m}ean_{j}ammy(3,:));$
figure(7)
zero$_{m}ean_{m}ax(1,:) = sum_{f}ourier_{m}ax_{p}lot - mean_{m}ax(1,1);$
zero$_{m}ean_{m}ax(2,:) = mean_{f}ourier_{m}ax_{p}lot - mean_{m}ax(1,2);$
zero$_{m}ean_{m}ax(3,:) = spread_{f}ourier_{m}ax_{p}lot - mean_{m}ax(1,3);$
scatter3(zero$_{m}ean_{m}ax(1,:), zero_{m}ean_{m}ax(2,:), zero_{m}ean_{m}ax(3,:),' black')axisvis3d$
covarience$_{s}um_{m}ax = cov(zero_{m}ean_{m}ax(1,:));$
covariance$_{m}ean_{m}ax = cov(zero_{m}ean_{m}ax(2,:));$
covariance$_{s}pread_{m}ax = cov(zero_{m}ean_{m}ax(3,:));$
figure(8)
[U$_{o}tis, S_{o}tis, V_{o}tis] = svd(zero_{m}ean_{o}tis);$
[m$_{o}tis, n_{o}tis] = size(zero_{m}ean_{o}tis);$
S2$_{o}tis = S_{o}tis(1 : m_{o}tis, 1 : m_{o}tis); Values_{o}tis = S2_{o}tis^{2}/n_{o}tis;$
Vectors$_{o}tis = U_{o}tis;$
Amplitudes$_{o}tis = S_{o}tis * conj(V_{o}tis');$
scatter$_{o}tis = [zero_{m}ean_{o}tis(1,:) + mean_{o}tis(1,1); zero_{m}ean_{o}tis(2,:) + mean_{o}tis(1,2); zero_{m}ean_{o}tis(3,$
$) + mean_{o}tis(1,3)];$
scatter3(scatter$_{o}tis(1,:), scatter_{o}tis(2,:), scatter_{o}tis(3,:),' b.'); holdonaxisvis3dtitle('TrainData');$
line1$_{o}tis = [(Values_{o}tis(1,1) * [0; Vectors_{o}tis(1,1)] + mean_{o}tis(1,1))'; (Values_{o}tis(1,1) *$
$[0; Vectors_{o}tis(2,1)] + mean_{o}tis(1,2))'; (Values_{o}tis(1,1) * [0; Vectors_{o}tis(3,1)] + mean_{o}tis(1,3))'];$
r1$_{o}tis = [line1_{o}tis(1,2) - line1_{o}tis(1,1), line1_{o}tis(2,2) - line1_{o}tis(2,1), line1_{o}tis(3,2) -$
$line1_{o}tis(3,1)];$
r$_{o}tis = abs(r1_{o}tis);$

p$_o tis(:,1) = mean_o tis' - 0.5 * r_o tis'$;

p$_o tis(:,2) = mean_o tis' + 0.5 * r_o tis'$;

plot3(p$_o tis(1,:), p_o tis(2,:), p_o tis(3,:),' b')$

[U$_j ammy, S_j ammy, V_j ammy] = svd(zero_m ean_j ammy)$;

[m$_j ammy, n_j ammy] = size(zero_m ean_j ammy)$;

S2$_j ammy = S_j ammy(1 : m_j ammy, 1 : m_j ammy); Values_j ammy = S2_j ammy^2 / n_j ammy$;

Vectors$_j ammy = U_j ammy$;

Amplitudes$_j ammy = S_j ammy * conj(V_j ammy')$;

scatter$_j ammy = [zero_m ean_j ammy(1,:) + mean_j ammy(1,1); zero_m ean_j ammy(2,:) + mean_j ammy(1,2); zero_m ean_j ammy(3,:) + mean_j ammy(1,3)]$;

scatter3(scatter$_j ammy(1,:), scatter_j ammy(2,:), scatter_j ammy(3,:),' r.')$;

line1$_j ammy = [(Values_j ammy(1,1) * [0; Vectors_j ammy(1,1)] + mean_j ammy(1,1))'; (Values_j ammy(1$ $[0; Vectors_j ammy(2,1)] + mean_j ammy(1,2))'; (Values_j ammy(1,1) * [0; Vectors_j ammy(3,1)] + mean_j ammy(1,3))']$;

r1$_j ammy = [line1_j ammy(1,2) - line1_j ammy(1,1), line1_j ammy(2,2) - line1_j ammy(2,1), line1_j ammy$ $line1_j ammy(3,1)]$;

r$_j ammy = abs(r1_j ammy)$;

p$_j ammy(:,1) = mean_j ammy' - r_j ammy'$;

p$_j ammy(:,2) = mean_j ammy' + r_j ammy'$;

plot3(p$_j ammy(1,:), p_j ammy(2,:), p_j ammy(3,:),' r')$

[U$_m ax, S_m ax, V_m ax] = svd(zero_m ean_m ax)$;

[m$_m ax, n_m ax] = size(zero_m ean_m ax)$;

S2$_m ax = S_m ax(1 : m_m ax, 1 : m_m ax); Values_m ax = S2_m ax^2 / n_m ax$;

Vectors$_m ax = U_m ax$;

Amplitudes$_m ax = S_m ax * conj(V_m ax')$;

scatter$_m ax = [zero_m ean_m ax(1,:) + mean_m ax(1,1); zero_m ean_m ax(2,:) + mean_m ax(1,2); zero_m ean_m ax($ $) + mean_m ax(1,3)]$;

scatter3(scatter$_m ax(1,:), scatter_m ax(2,:), scatter_m ax(3,:),' k.')$;

line1$_m ax = [(Values_m ax(1,1) * [0; Vectors_m ax(1,1)] + mean_m ax(1,1))'; (Values_m ax(1,1) * [0; Vectors_m ax(2,1)] + mean_m ax(1,2))'; (Values_m ax(1,1) * [0; Vectors_m ax(3,1)] + mean_m ax(1,3))']$;

line2$_m ax = [(Values_m ax(1,1) * [0; Vectors_m ax(1,2)] + mean_m ax(1,1))'; (Values_m ax(1,1) * [0; Vectors_m ax(2,2)] + mean_m ax(1,2))'; (Values_m ax(1,1) * [0; Vectors_m ax(3,2)] + mean_m ax(1,3))']$;

r1$_m ax = [line1_m ax(1,2) - line1_m ax(1,1), line1_m ax(2,2) - line1_m ax(2,1), line1_m ax(3,2) - line1_m ax(3,1)]$;

r2$_m ax = [line2_m ax(1,2) - line2_m ax(1,1), line2_m ax(2,2) - line2_m ax(2,1), line2_m ax(3,2) - line2_m ax(3,1)]$;

r$_m ax = abs(r1_m ax)$;

p$_m ax(:,1) = mean_m ax' - 1.5 * r_m ax'$;

p$_m ax(:,2) = mean_m ax' + r_m ax'$;

plot3(p$_m ax(1,:), p_m ax(2,:), p_m ax(3,:),' k')$

weight1 = 10;

weight2 = 1;

d$_o tis_right = point_to_line_distance(scatter_otis', p_otis(:,1)', p_otis(:,2)');$

for k = 1: length(d$_o tis_r ight$)

d$_o tis_right(k,:) = weight1*d_otis_right(k,:) + weight2*norm(abs(scatter_otis(:,k)' - mean_otis));$

end

d$_o tis_wrong1 = point_to_line_distance(scatter_jammy', p_otis(:,1)', p_otis(:,2)');$

d$_o tis_wrong2 = point_to_line_distance(scatter_max', p_otis(:,1)', p_otis(:,2)');$

d$_o tis_wrong = zeros(length(d_otis_wrong1) + length(d_otis_wrong2), 1);$

for k = 1 : length(d$_o tis_w rong1$)

d$_o tis_wrong(k,1) = weight1*d_otis_wrong1(k) + weight2*norm(abs(scatter_jammy(:,k)' - mean_otis));$

end

for k = 1 : k

d$_o tis_wrong(k + length(d_otis_wrong1), 1) = weight1*d_otis_wrong2(k) + weight2*norm(abs(scatter_max(:,k)' - mean_otis));$

end

sector$_o tis = max(d_otis_right);$

next$_s ector_otis = min(d_otis_wrong);$

d$_j ammy_right = point_to_line_distance(scatter_jammy', p_jammy(:,1)', p_jammy(:,2)');$

for k = 1: length(d$_j ammy_r ight$)

d$_j ammy_right(k,:) = weight1*d_jammy_right(k,:) + weight2*norm(abs(scatter_jammy(:,k)' - mean_jammy));$

end

d$_j ammy_wrong1 = point_to_line_distance(scatter_otis', p_jammy(:,1)', p_jammy(:,2)');$

d$_j ammy_wrong2 = point_to_line_distance(scatter_max', p_jammy(:,1)', p_jammy(:,2)');$

d$_j ammy_wrong = zeros(length(d_jammy_wrong1) + length(d_jammy_wrong2), 1);$

for k = 1 : length(d$_j ammy_w rong1$)

d$_j ammy_wrong(k,1) = weight1*d_jammy_wrong1(k) + weight2*norm(abs(scatter_otis(:,k)' - mean_jammy));$

end

for k = 1 : k

d$_j ammy_wrong(k + length(d_jammy_wrong1), 1) = weight1*d_jammy_wrong2(k) + weight2*norm(abs(scatter_max(:,k)' - mean_jammy));$

end

sector$_j ammy = max(d_jammy_right);$

next$_s ector_jammy = min(d_jammy_wrong);$

d$_m ax_right = point_to_line_distance(scatter_max', p_max(:,1)', p_max(:,2)');$

for k = 1 : length(d$_m ax_r ight$)

d$_m ax_right(k,:) = weight1*d_max_right(k,:) + weight2*norm(abs(scatter_max(:,k)' - mean_max));$

end

d$_{m}ax_{w}rong1 = point_{t}o_{l}ine_{d}istance(scatter_{o}tis', p_{m}ax(:,1)', p_{m}ax(:,2)');$

d$_{m}ax_{w}rong2 = point_{t}o_{l}ine_{d}istance(scatter_{j}ammy', p_{m}ax(:,1)', p_{m}ax(:,2)');$

d$_{m}ax_{w}rong = zeros(length(d_{m}ax_{w}rong1) + length(d_{m}ax_{w}rong2), 1);$

for k = 1 : length(d$_{m}ax_{w}rong1)$

d$_{m}ax_{w}rong(k,1) = weight1 * d_{m}ax_{w}rong1(k) + weight2 * norm(abs(scatter_{o}tis(:,k)' - mean_{m}ax));$

end

for k = 1 : k

d$_{m}ax_{w}rong(k + length(d_{m}ax_{w}rong1), 1) = weight1 * d_{m}ax_{w}rong2(k) + weight2 * norm(abs(scatter_{j}ammy(:,k)' - mean_{m}ax));$

end

sector$_{m}ax = max(d_{m}ax_{r}ight);$

next$_{s}ector_{m}ax = min(d_{m}ax_{w}rong);$

next$_{s}ector = [next_{s}ector_{o}tis, next_{s}ector_{j}ammy, next_{s}ector_{m}ax];$

border$_{o}tis = (sector_{o}tis + next_{s}ector_{o}tis);$

border$_{j}ammy = (sector_{j}ammy + next_{s}ector_{j}ammy)/2;$

border$_{m}ax = (sector_{m}ax + next_{s}ector_{m}ax);$

samples$_{p}er_{s}ong = 10;$

number$_{o}f_{s}ongs = 6;$

number$_{o}f_{s}amples_{p}er_{g}enre = samples_{p}er_{s}ong * 2;$

number$_{o}f_{s}amples = samples_{p}er_{s}ong * number_{o}f_{s}ongs;$

sample = cell(samples$_{p}er_{s}ong, 1);$

for k = 1:samples$_{p}er_{s}ong$

samplek = [2000000+((k-1)*225000) 2000000+(k*225000)];

end

y$_{t}est = cell(number_{o}f_{s}amples); Fs_{t}est = cell(number_{o}f_{s}amples);$

for k = 1:samples$_{p}er_{s}ong$

[y$_{t}estk, Fs_{t}estk] = audioread(''Complicate_{y}a(hiphop).mp3'', samplek);$

[y$_{t}estk + 1 * samples_{p}er_{s}ong, Fs_{t}estk + 1 * samples_{p}er_{s}ong] = audioread(''Here_{I}f_{Y}ou_{r}e_{G}oing(hipho$

[y$_{t}estk + 2 * samples_{p}er_{s}ong, Fs_{t}estk + 2 * samples_{p}er_{s}ong] = audioread(''Present_{D}ay(jazz).mp3'', s$

[y$_{t}estk + 3 * samples_{p}er_{s}ong, Fs_{t}estk + 3 * samples_{p}er_{s}ong] = audioread(''Tango_{B}ango(jazz).mp3'',$

[y$_{t}estk + 4 * samples_{p}er_{s}ong, Fs_{t}estk + 4 * samples_{p}er_{s}ong] = audioread(''Cosmic_{L}ove(Electro).mp3$

[y$_{t}estk + 5 * samples_{p}er_{s}ong, Fs_{t}estk + 5 * samples_{p}er_{s}ong] = audioread(''Tired_{B}reak(Electro).mp3'$

end

signal$_{t}est = cell(number_{o}f_{s}amples, 1);$

for k = 1:number$_{o}f_{s}amples$

signal$_{t}estk = y_{t}estk(:,1)';$

end

data$_{p}oints_{t}est = cell(number_{o}f_{s}amples, 1); time_{t}est = cell(number_{o}f_{s}amples, 1);$

for k = 1:number$_{o}f_{s}amples$

data$_{p}oints_{t}estk = length(signal_{t}estk); time_{t}estk = data_{p}oints_{t}estk/Fs_{t}estk;$
end
data$_{v}ector_{t}est = cell(number_{o}f_{s}amples, 1); time_{v}ector_{t}est = cell(number_{o}f_{s}amples, 1);$
for k = 1:number$_{o}f_{s}amples$
data$_{v}ector_{t}estk = 1 : data_{p}oints_{t}estk; time_{v}ector_{t}estk = data_{v}ector_{t}estk/Fs_{t}estk;$
end
fft$_{t}est = cell(number_{o}f_{s}amples, 1); frequencies_{s}pace_{t}est = cell(number_{o}f_{s}amples, 1); frequencies_{s}pa$
$cell(number_{o}f_{s}amples, 1);$
for k = 1:number$_{o}f_{s}amples$
fft$_{t}estk = fft(signal_{t}estk);$
frequencies$_{s}pace_{t}estk = (2 * pi/time_{t}estk * [0 : (data_{p}oints_{t}estk/2) - data_{p}oints_{t}estk/2 :$
$-1]); frequencies_{s}pace_{s}hifted_{t}estk = fftshift(frequencies_{s}pace_{t}estk);$
end
fourier$_{t}est = cell(number_{o}f_{s}amples, 1);$
spread$_{f}actor = 1000;$
spread$_{v}ariable = 5;$
for k = 1:number$_{o}f_{s}amples$
fourier$_{t}estk(1, :) = frequencies_{s}pace_{s}hifted_{t}estk; fourier_{t}estk(2, :) = fftshift(abs(fft_{t}estk));$
end
length$_{f}ourier_{t}est = cell(number_{o}f_{s}amples, 1);$
sum$_{f}ourier_{t}est = cell(number_{o}f_{s}amples, 1);$
max$_{f}ourier_{t}est = cell(number_{o}f_{s}amples, 1);$
mean$_{f}ourier_{t}est = cell(number_{o}f_{s}amples, 1);$
spread$_{f}ourier_{t}est = cell(number_{o}f_{s}amples, 1);$
for j = 1 : number$_{o}f_{s}amples$
length$_{f}ourier_{t}estj = length(fourier_{t}estj);$
max$_{f}ourier_{t}estj = max(abs(fourier_{t}estj(1, j)));$
sum$_{f}ourier_{t}estj = 0;$
mean$_{f}ourier_{t}estj = 0;$
spread$_{f}ourier_{t}estj = 0;$
for k = 1 : length$_{f}ourier_{t}estj$
sum$_{f}ourier_{t}estj = sum_{f}ourier_{t}estj + fourier_{t}estj(2, k);$
mean$_{f}ourier_{t}estj = mean_{f}ourier_{t}estj + abs(fourier_{t}estj(1, k)) * fourier_{t}estj(2, k)/max_{f}ourier_{t}estj;$
if fourier$_{t}estj(2, k) > (max_{f}ourier_{t}estj/spread_{f}actor)$
spread$_{f}ourier_{t}estj = spread_{f}ourier_{t}estj + 1;$
end
end
sum$_{f}ourier_{t}estj = sum_{f}ourier_{t}estj/length_{f}ourier_{t}estj;$
mean$_{f}ourier_{t}estj = mean_{f}ourier_{t}estj/length_{f}ourier_{t}estj;$
spread$_{f}ourier_{t}estj = (spread_{f}ourier_{t}estj)^{(}1/spread_{v}ariable);$
end

figure(9)

plot3(p$_otis(1,:), p_otis(2,:), p_otis(3,:), 'b')holdontitle('TestData')plot3(p_jammy(1,:), p_jammy(2,:), p_jammy(3,:), 'r')plot3(p_max(1,:), p_max(2,:), p_max(3,:), 'k')$

sum$_fourier_test_plot = zeros(1, number_of_samples);$

mean$_fourier_test_plot = zeros(1, number_of_samples);$

spread$_fourier_test_plot = zeros(1, number_of_samples);$

for k = 1:number$_of_samples$

sum$_fourier_test_plot(k) = sum_fourier_testk;$

mean$_fourier_test_plot(k) = mean_fourier_testk;$

spread$_fourier_test_plot(k) = spread_fourier_testk;$

end

for k = 1 : number$_of_samples_per_genre$

scatter3(sum$_fourier_test_plot(k), mean_fourier_test_plot(k), spread_fourier_test_plot(k), 'b.')$

end

for k = 1 : number$_of_samples_per_genre$

scatter3(sum$_fourier_test_plot(k + number_of_samples_per_genre), mean_fourier_test_plot(k + number_of_samples_per_genre), spread_fourier_test_plot(k + number_of_samples_per_genre), 'r.')$

end

for k = 1 : number$_of_samples_per_genre$

scatter3(sum$_fourier_test_plot(k + 2 * number_of_samples_per_genre), mean_fourier_test_plot(k + 2 * number_of_samples_per_genre), spread_fourier_test_plot(k + 2 * number_of_samples_per_genre), 'k.')$

end

axis vis3d

scatter$_hiphop = zeros(3, number_of_samples_per_genre);$

scatter$_jazz = zeros(3, number_of_samples_per_genre);$

scatter$_electro = zeros(3, number_of_samples_per_genre);$

for k = 1 : number$_of_samples_per_genre$

scatter$_hiphop(:, k) = [sum_fourier_test_plot(k); mean_fourier_test_plot(k); spread_fourier_test_plot(k)];$

end

for k = 1 : number$_of_samples_per_genre$

scatter$_jazz(:, k) = [sum_fourier_test_plot(k + number_of_samples_per_genre); mean_fourier_test_plot(k + number_of_samples_per_genre); spread_fourier_test_plot(k + number_of_samples_per_genre)];$

end

for k = 1 : number$_of_samples_per_genre$

scatter$_electro(:, k) = [sum_fourier_test_plot(k + 2 * number_of_samples_per_genre); mean_fourier_test_plot(k + 2 * number_of_samples_per_genre); spread_fourier_test_plot(k + 2 * number_of_samples_per_genre)];$

end

hiphop$_right = 0;$

hiphop$_wrong = 0;$

not$_hiphop_right = 0;$

not$_hiphop_wrong = 0;$

for k = 1 : number$_{o}f_{s}amples_{p}er_{g}enre$
d$_{o}tis_{r}ight(k,:) = weight1 * point_{t}o_{l}ine_{d}istance(scatter_{h}iphop(:,k)', p_{o}tis(:,1)', p_{o}tis(:,2)') + weight2 * norm(abs(scatter_{h}iphop(:,k)' - mean_{o}tis));$
if d$_{o}tis_{r}ight(k,:) <= border_{o}tis$
hiphop$_{r}ight = hiphop_{r}ight + 1;$
else
hiphop$_{w}rong = hiphop_{w}rong + 1;$
end
d$_{o}tis_{w}rong1(k,:) = weight1 * point_{t}o_{l}ine_{d}istance(scatter_{j}azz(:,k)', p_{o}tis(:,1)', p_{o}tis(:,2)') + weight2 * norm(abs(scatter_{j}azz(:,k)' - mean_{o}tis));$
if d$_{o}tis_{w}rong1(k,:) > border_{o}tis$
not$_{h}iphop_{r}ight = not_{h}iphop_{r}ight + 1;$
else
not$_{h}iphop_{w}rong = not_{h}iphop_{w}rong + 1;$
end
d$_{o}tis_{w}rong2(k,:) = weight1 * point_{t}o_{l}ine_{d}istance(scatter_{e}lectro(:,k)', p_{o}tis(:,1)', p_{o}tis(:,2)') + weight2 * norm(abs(scatter_{e}lectro(:,k)' - mean_{o}tis));$
if d$_{o}tis_{w}rong2(k,:) > border_{o}tis$
not$_{h}iphop_{r}ight = not_{h}iphop_{r}ight + 1;$
else
not$_{h}iphop_{w}rong = not_{h}iphop_{w}rong + 1;$
end
end
jazz$_{r}ight = 0;$
jazz$_{w}rong = 0;$
not$_{j}azz_{r}ight = 0;$
not$_{j}azz_{w}rong = 0;$
for k = 1 : number$_{o}f_{s}amples_{p}er_{g}enre$
d$_{j}ammy_{r}ight(k,:) = weight1 * point_{t}o_{l}ine_{d}istance(scatter_{j}azz(:,k)', p_{j}ammy(:,1)', p_{j}ammy(:,2)') + weight2 * norm(abs(scatter_{j}azz(:,k)' - mean_{j}ammy));$
if d$_{j}ammy_{r}ight(k,:) <= border_{j}ammy$
jazz$_{r}ight = jazz_{r}ight + 1;$
else
jazz$_{w}rong = jazz_{w}rong + 1;$
end
d$_{j}ammy_{w}rong1(k,:) = weight1 * point_{t}o_{l}ine_{d}istance(scatter_{h}iphop(:,k)', p_{j}ammy(:,1)', p_{j}ammy(:,2)') + weight2 * norm(abs(scatter_{h}iphop(:,k)' - mean_{j}ammy));$
if d$_{j}ammy_{w}rong1(k,:) > border_{j}ammy$
not$_{j}azz_{r}ight = not_{j}azz_{r}ight + 1;$
else
not$_{j}azz_{w}rong = not_{j}azz_{w}rong + 1;$

end

d$_{j}ammy_{w}rong2(k,:) = weight1 * point_{t}o_{l}ine_{d}istance(scatter_{e}lectro(:,k)', p_{j}ammy(:,1)', p_{j}ammy(:,2)') + weight2 * norm(abs(scatter_{e}lectro(:,k)' - mean_{j}ammy));$

if d$_{j}ammy_{w}rong2(k,:) > border_{j}ammy$

not$_{j}azz_{r}ight = not_{j}azz_{r}ight + 1;$

else

not$_{j}azz_{w}rong = not_{j}azz_{w}rong + 1;$

end

end

electro$_{r}ight = 0;$

electro$_{w}rong = 0;$

not$_{e}lectro_{r}ight = 0;$

not$_{e}lectro_{w}rong = 0;$

for k = 1 : number$_{o}f_{s}amples_{p}er_{g}enre$

d$_{m}ax_{r}ight(k,:) = weight1 * point_{t}o_{l}ine_{d}istance(scatter_{e}lectro(:,k)', p_{m}ax(:,1)', p_{m}ax(:,2)') + weight2 * norm(abs(scatter_{e}lectro(:,k)' - mean_{m}ax));$

if d$_{m}ax_{r}ight(k,:) <= border_{m}ax$

electro$_{r}ight = electro_{r}ight + 1;$

else

electro$_{w}rong = electro_{w}rong + 1;$

end

d$_{m}ax_{w}rong1(k,:) = weight1 * point_{t}o_{l}ine_{d}istance(scatter_{h}iphop(:,k)', p_{m}ax(:,1)', p_{m}ax(:,2)') + weight2 * norm(abs(scatter_{h}iphop(:,k)' - mean_{m}ax));$

if d$_{m}ax_{w}rong1(k,:) > border_{m}ax$

not$_{e}lectro_{r}ight = not_{e}lectro_{r}ight + 1;$

else

not$_{e}lectro_{w}rong = not_{e}lectro_{w}rong + 1;$

end

d$_{m}ax_{w}rong2(k,:) = weight1 * point_{t}o_{l}ine_{d}istance(scatter_{j}azz(:,k)', p_{m}ax(:,1)', p_{m}ax(:,2)') + weight2 * norm(abs(scatter_{j}azz(:,k)' - mean_{m}ax));$

if d$_{m}ax_{w}rong2(k,:) > border_{m}ax$

not$_{e}lectro_{r}ight = not_{e}lectro_{r}ight + 1;$

else

not$_{e}lectro_{w}rong = not_{e}lectro_{w}rong + 1;$

end

end

quote$_{h}iphop_{r}ight = hiphop_{r}ight / number_{o}f_{s}amples_{p}er_{g}enre$

quote$_{n}ot_{h}iphop_{r}ight = not_{h}iphop_{r}ight / (2 * number_{o}f_{s}amples_{p}er_{g}enre)$

quote$_{j}azz_{r}ight = jazz_{r}ight / number_{o}f_{s}amples_{p}er_{g}enre$

quote$_{n}ot_{j}azz_{r}ight = not_{j}azz_{r}ight / (2 * number_{o}f_{s}amples_{p}er_{g}enre)$

quote$_{e}lectro_{r}ight = electro_{r}ight / number_{o}f_{s}amples_{p}er_{g}enre$

quote$_n ot_e lectro_r ight = not_e lectro_r ight/(2 * number_o f_s amples_p er_g enre)$

function distance=point$_t o_l ine_d istance(pt, v1, v2)$

if nargin =3 error('HJW:point$_t o_l ine_d istance : nargin', ...'Incorrectnumberofinputs, expected3.'); endi$
[23])$||any(size(pt) == 0)error('HJW : point_t o_l ine_d istance : pt_t ype_s ize', ...'Firstinput(pt)isnotnumeric$
$size(pt, 2)error('HJW : point_t o_l ine_d istance : v_t ype_s ize', ...['Secondinput(v1)isnotnumericorhasanincorre$
$size(pt, 2)error('HJW : point_t o_l ine_d istance : v_t ype_s ize', ['Thirdinput(v2)', ...'isnotnumericorhasanincorr$
$v1(:)'; iflength(v1) == 2, v1(3) = 0; endv2 = v2(:)'; iflength(v2) == 2, v2(3) =$
$0; endifsize(pt, 2) == 2, pt(1, 3) = 0; endv1_= repmat(v1, size(pt, 1), 1); v2_= repmat(v2, size(pt, 1), 1);$

a = v1$_- v2_; b = pt - v2_; distance = sqrt(sum(cross(a, b, 2).^2, 2))./sqrt(sum(a.^2, 2));$

end