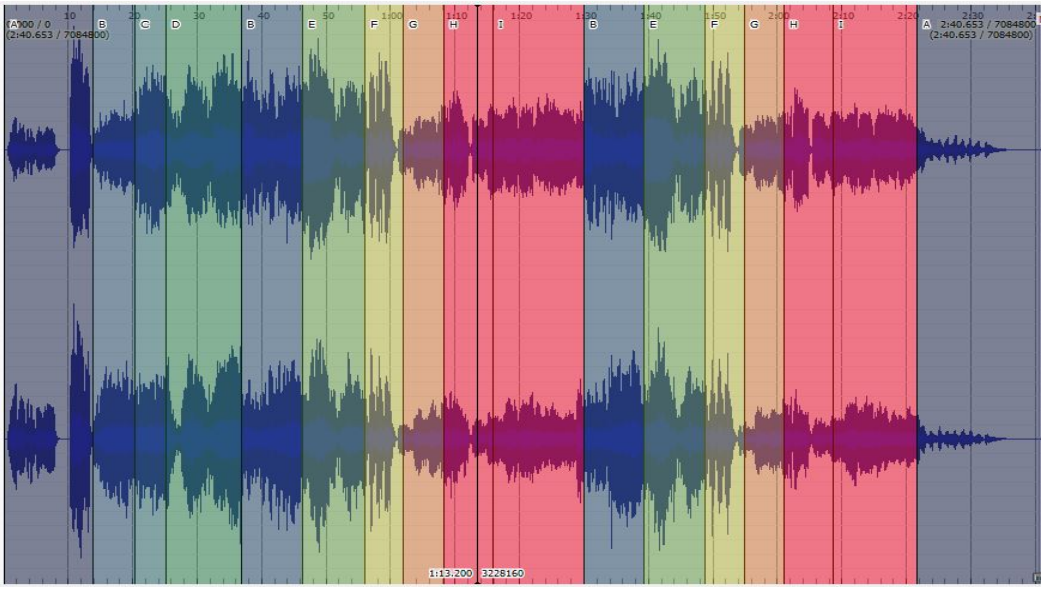
Hector Perez V00794415 Assignment 5 CSC 475

Vamp Plugins and MirEval

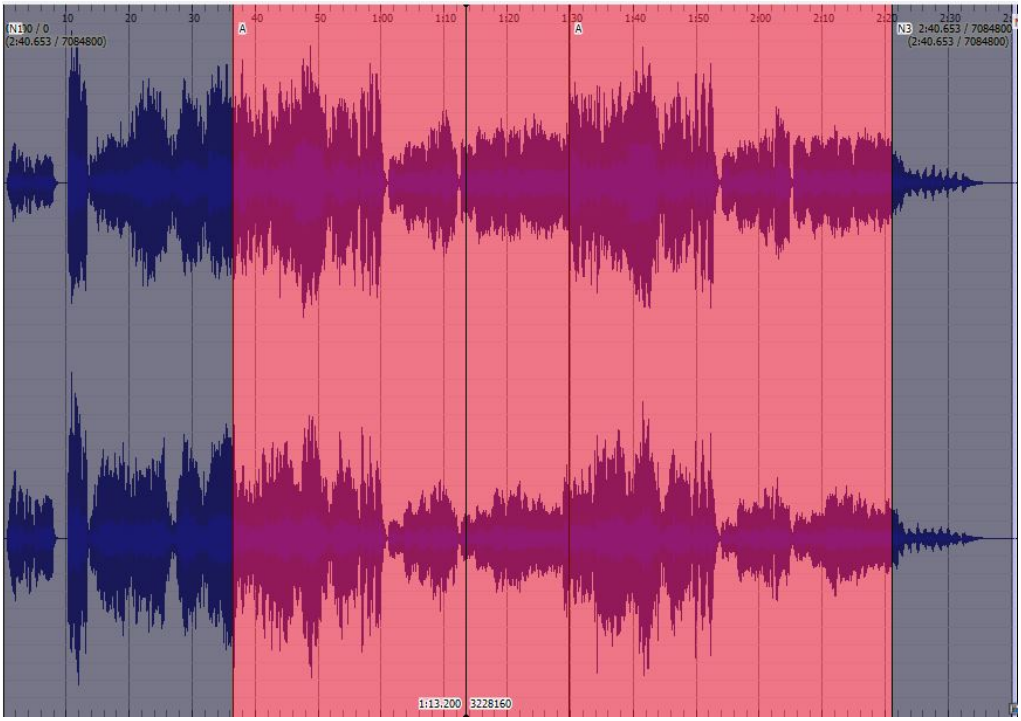
QUESTION 1:

Piece: *Gloria in Excelsis Deo* from Mozart's Mass in C Minor. Duration ~ 3 Min.

Sonic Visualizer - Using Segmenter: (Recognized 9 different segments)

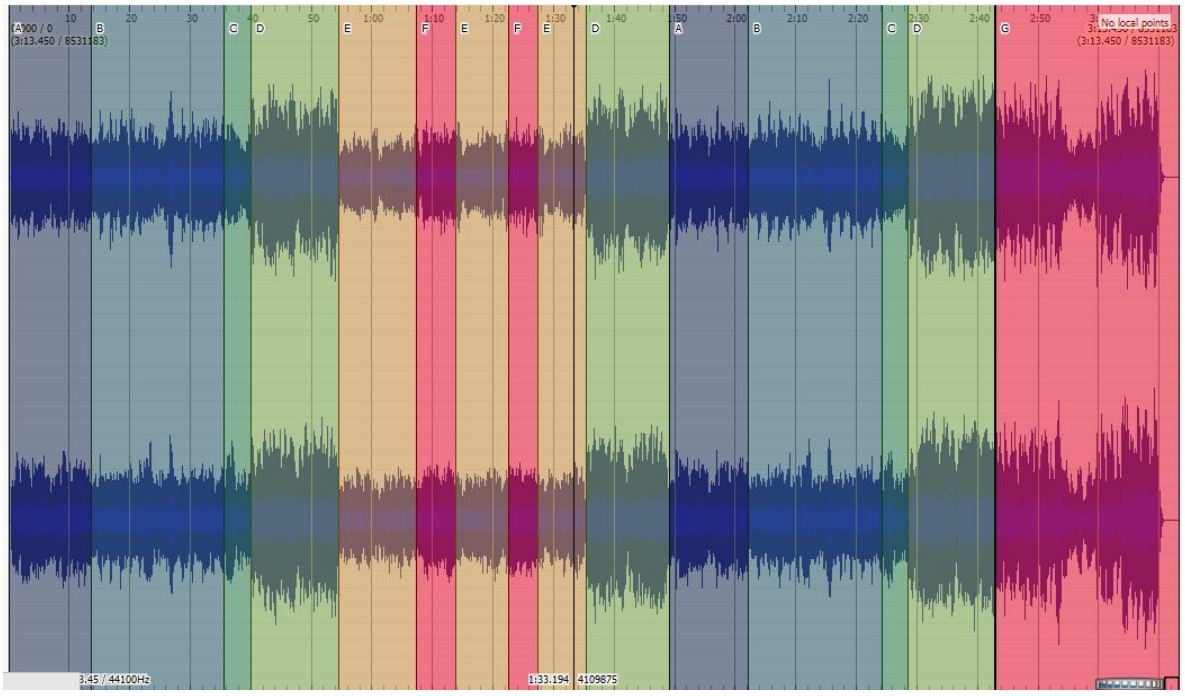


Sonic Visualizer - Using Segmentino (4 segments found - 1 characteristic).

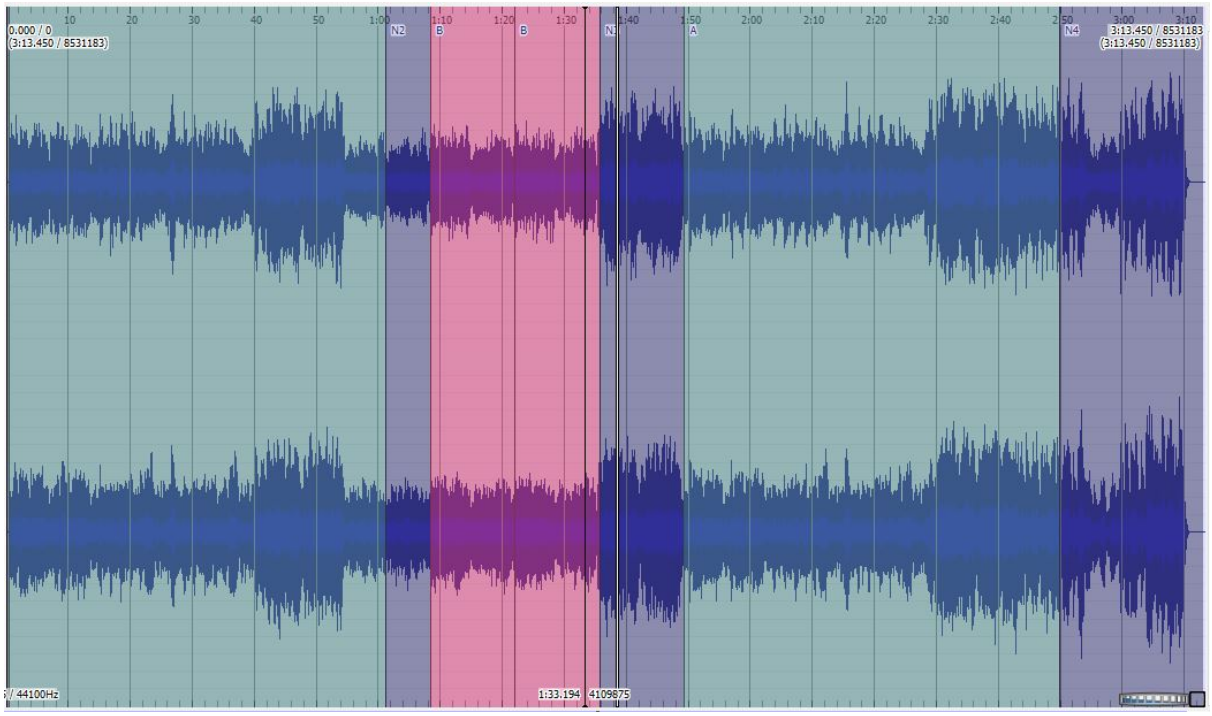


Piece: *Rondo Alla Turca* - Mozart ~ 3 Minutes.

Sonic Visualizer - Using Segmenter. (7 segments found)



Sonic Visualizer - Using Segmentino. (6 segments found - 3 are characteristic)



Processing the same tracks with Sonic Annotator from the command line :

I learned that the command has to be composed of the transform to be used, the audio files to be processed and the write module to be used (.csv, .lib, stdout etc). There is a -s command and a -d command. The -d command just runs the program with default settings. -s is skeleton and shows RDF file with settings that the transform uses. In this case, running with -s produced no output for me, so I just ran it with the -d option.

Processing *Gloria in Excelsis Deo* with Segmenter.
Command : sonic-annotator -d vamp:qm-vamp-plugins:qm-segmenter:segmentation gloria.mp3 -w csv --csv-stdout

:

```
Extracting features for: "gloria.mp3"
Audio file "gloria.mp3": 2ch at 44100Hz
Extracting and writing features... Done
"gloria.mp3",0.000000000,13.800000000,1,"A"
,13.800000000,6.200000000,2,"B"
,20.000000000,7.400000000,3,"C"
,27.400000000,10.600000000,4,"D"
,38.000000000,8.800000000,5,"E"
,46.800000000,14.200000000,6,"F"
,61.000000000,5.400000000,7,"G"
,66.400000000,5.800000000,8,"H"
,72.200000000,6.200000000,7,"G"
,78.400000000,7.400000000,9,"I"
,85.800000000,4.600000000,10,"J"
,90.400000000,9.200000000,5,"E"
,99.600000000,14.200000000,6,"F"
,113.800000000,5.400000000,7,"G"
,119.200000000,5.800000000,8,"H"
,125.000000000,6.200000000,7,"G"
,131.200000000,7.400000000,9,"I"
,138.600000000,22.000000000,10,"J"
```

Processing *Gloria in Excelsis Deo* with Segmentino.
Command: sonic-annotator -d vamp:segmentino:segmentino:segmentation gloria.mp3 -w csv --csv-stdout

```
Extracting and writing features... Done
"gloria.mp3",0.000000000,36.594648526,0,"N1"
,36.594648526,53.150476190,1,"A"
,89.745124716,51.281269841,1,"A"
,141.026394557,19.110022676,0,"N3"
```

Continue below

Processing *Rondo Alla Turca* with Segmenter.
Command: sonic-annotator -d vamp:qm-vamp-plugins:qm-segmenter:segmentation turca.mp3 -w csv --csv-stdout

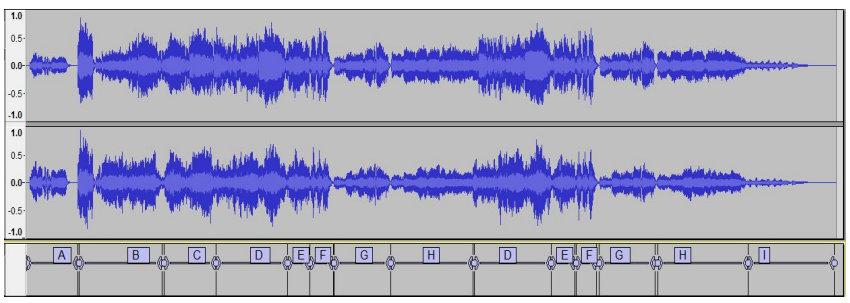
```
Extracting and writing features... Done
"turca.mp3",0.000000000,13.600000000,1,"A"
,13.600000000,8.200000000,2,"B"
,21.800000000,4.800000000,3,"C"
,26.600000000,8.800000000,2,"B"
,35.400000000,4.800000000,3,"C"
,40.200000000,14.000000000,4,"D"
,54.200000000,13.200000000,5,"E"
,67.400000000,7.000000000,6,"F"
,74.400000000,6.800000000,5,"E"
,81.200000000,6.800000000,6,"F"
,88.000000000,6.800000000,5,"E"
,94.800000000,14.400000000,4,"D"
,109.200000000,13.200000000,1,"A"
,122.400000000,8.200000000,2,"B"
,130.600000000,4.600000000,3,"C"
,135.200000000,9.000000000,2,"B"
,144.200000000,4.600000000,3,"C"
,148.800000000,14.200000000,4,"D"
,163.000000000,30.000000000,7,"G"
```

Processing *Rondo Alla Turca* with Segmentino.
Command: sonic-annotator -d vamp:segmentino:segmentino:segmentation turca.mp3 -w csv --csv-stdout

```
Extracting and writing features... Done
"turca.mp3",0.000000000,0.406349206,0,"N1"
,0.406349206,60.894331066,1,"A"
,61.300680272,7.209795918,0,"N2"
,68.510476190,13.560453515,2,"B"
,82.070929705,13.688163265,2,"B"
,95.759092970,13.560453515,0,"N3"
,109.319546485,60.638911564,1,"A"
,169.958458049,22.685895692,0,"N4"
```

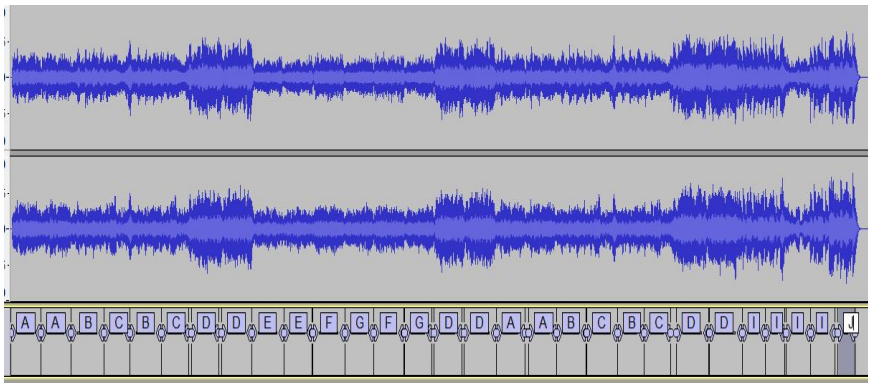
Annotating the structure on Audacity:

Labels for Gloria in Excelsis Deo: (I had 9 segments)



0.000000	10.191018	A
10.459202	26.952560	B
27.354837	37.679947	C
37.679947	51.893735	D
51.893735	56.318782	E
56.318782	61.012014	F
61.146106	72.409863	G
72.409863	88.769128	H
89.037312	104.323839	D
104.323839	109.017071	E
109.285255	113.308026	F
113.844395	124.974059	G
125.510428	143.478802	H
143.478802	160.642621	I

Labels for Rondo Alla Turca: (I had 10 segments)



0.000000	6.765562	A
6.765562	13.531124	A
13.531124	20.941025	B
20.941025	26.740079	C
26.740079	33.827810	B
33.827810	39.949033	C
40.593372	46.714595	D
47.358934	54.124496	D
54.124496	61.373313	E
61.373313	67.655620	E
67.816705	75.065522	F
75.065522	81.508914	G
81.508914	88.274476	F
88.435561	94.556784	G
95.040038	101.322346	D
101.805600	109.054417	D
109.054417	115.497809	A
116.303233	122.585541	A
122.585541	129.190018	B
129.351103	136.277749	C
136.277749	142.237887	B
142.237887	148.198025	C
149.486704	156.735520	D
156.896605	164.306506	D
164.306506	169.461220	I

Discussion of similarities and differences:

Gloria in Excelsis Deo

A) Differences between Segmenter's segments and my segments. (Audio-Wise).
The segmentation was pretty similar visually, and also audibly. I was being careful to pick segments based on themes in the piece. Segmenter did a good job of picking out the differences. There is a problem with one of the segments, but it's understandable. The first and last segments of Segmenter's output sounded completely different to each other, but they're both labeled as A.

B) Differences between Segmentino's segments and my segments. (Audio-Wise).
The segments generated are really large, and do serve to get a general idea of the piece. However, these segments were really different from mine and lasted much longer.

Rondo Alla Turca

C) Differences between Segmenter's segments and my segments. (Audio-Wise).
I think I had a bit of a bias for this piece, because I know the structure better - it's more familiar. It seems I picked out a lot of smaller segments. Segmenter again did a good job, but still had sections that would have contained two or three of my smaller segments.

D) Differences between Segmentino's segments and my segments. (Audio-Wise).
Segmentino again generated segments that are too large to objectively compare against mine.

Continue below

Question 2 - MIR Eval - Evaluation of the segmentations.

Note: Prior to finding that mir_eval has a io.load method, I spent some time figuring out how to import CSV's so I decide to include that code here.

Code: (Without using io.load method)

workAround.py

```
Must import auxFunctions, import numpy as np ,import mir_eval,import csv.

timesSegmentinoGloria = np.array(auxFunctions.getTimesFromCsv('segmentino_gloria.csv'));
timesSegmentinoTurca = np.array(auxFunctions.getTimesFromCsv('segmentino_turca.csv'));
timesSegmenterGloria = np.array(auxFunctions.getTimesFromCsv('segmenter_gloria.csv'));
timesSegmenterTurca = np.array(auxFunctions.getTimesFromCsv('segmenter_turca.csv'));
timesHectorGloria = np.array(auxFunctions.getTimesFromCsv('hector_gloria.csv'));
timesHectorTurca = np.array(auxFunctions.getTimesFromCsv('hector_turca.csv'));
print "-----No.1 - Gloria in Excelsis Deo ----------------------------------------------"
f_measureSegmentinoGloria =
auxFunctions.getF_Measure(timesHectorGloria,timesSegmentinoGloria);
print("f_measure for Segmentino: %f" % f_measureSegmentinoGloria);

f_measureSegmenterGloria =
auxFunctions.getF_Measure(timesHectorGloria,timesSegmenterGloria);
print("f_measure for Segmenter : %f" % f_measureSegmenterGloria);

print "------No.2 - Rondo Alla Turca-------------------------------------------------------"

f_measureSegmentinoTurca = auxFunctions.getF_Measure(timesHectorTurca,timesSegmentinoTurca);
print("f_measure for Segmentino: %f" % f_measureSegmentinoTurca);

f_measureSegmenterTurca = auxFunctions.getF_Measure(timesHectorTurca,timesSegmenterTurca);
print("f_measure for Segmenter: %f" % f_measureSegmenterTurca);
```

auxFunctions.py

```
Must import csv,import mir_eval,import numpy as np.

def getTimesFromCsv(filename):
	mode = 'r';
	output = [];
	with open(filename, mode) as f:
		reader = csv.DictReader(f);
		for row in reader:
			output.append([float(row['startTime']), float(row['endTime'])]);
	return output;

def getF_Measure(reference,estimated):
	p,r,f_measure = mir_eval.segment.detection(reference,estimated);
	return f_measure;
```

Code: Using io.load method from mir_eval.

workAround2.py

```
Must import mir_eval,import auxFunctions.

segmentino_gloria = mir_eval.io.load_intervals('segmentino_gloria_2.csv', delimiter=',');
segmenter_gloria = mir_eval.io.load_intervals('segmenter_gloria_2.csv', delimiter=',');
segmentino_turca = mir_eval.io.load_intervals('segmentino_turca_2.csv', delimiter=',');
segmenter_turca = mir_eval.io.load_intervals('segmenter_turca_2.csv', delimiter=',');
hector_gloria = mir_eval.io.load_intervals('hector_gloria_2.csv', delimiter=',');
hector_turca = mir_eval.io.load_intervals('hector_turca_2.csv', delimiter=',');

print "-----No.1 - Gloria in Excelsis Deo
------------------------------------------------"

f_measureSegmentinoGloria = auxFunctions.getF_Measure(hector_gloria,segmentino_gloria);
print("f_measure for Segmentino: %f" % f_measureSegmentinoGloria);

f_measureSegmenterGloria = auxFunctions.getF_Measure(hector_gloria,segmenter_gloria);
print("f_measure for Segmenter : %f" % f_measureSegmenterGloria);

print "------No.2 - Rondo Alla
Turca-------------------------------------------------------"

f_measureSegmentinoTurca = auxFunctions.getF_Measure(hector_turca,segmentino_turca);
print("f_measure for Segmentino: %f" % f_measureSegmentinoTurca);

f_measureSegmenterTurca = auxFunctions.getF_Measure(hector_turca,segmenter_turca);
print("f_measure for Segmenter: %f" % f_measureSegmenterTurca);
```

Output: (Same for both workAround1.py and workAround2.py)

```
-----No.1 - Gloria in Excelsis Deo ------------------------------------------------
f_measure for Segmentino: 0.074074
f_measure for Segmenter : 0.153846
------No.2 - Rondo Alla Turca-------------------------------------------------------
f_measure for Segmentino: 0.156863
f_measure for Segmenter: 0.354839
```

Discussion: Does the pair-wise F-measure correspond to your listening perception of segmentation quality?

a) F measure for *Gloria in Excelsis Deo* segmented with Segmentino:
Segmentino returned a low F-measure. It's segmentation wasn't close to the ground truth. This is really audible, because the segments segmentino returned were too large.
b) F measure for *Gloria in Excelsis Deo* segmented with Segmenter:
Segmenter achieved a slightly higher F-measure. Audibly it corresponds to my perception, because the segments were smaller, and identified shorter themes.

c) F measure for *Rondo Alla Turca* segmented with Segmentino:
Segmentino achieved a higher F-measure. I wasn't expecting that because again the segments were too large. Perhaps the segments' endpoints were closer to the labels I created, even though Segmentino returned a small number of segments,

d) F measure for *Rondo Alla Turca* segmented with Segmenter:
Segmenter performed better than Segmentino, and the segments it returned are audibly similar to the segments I selected manually by listening to the piece.

The F-measure is a measure that combines precision and recall. It is the average of the two when they are close (it is the weighted average of the two measures).

Precision is $\frac{tp}{tp+fp}$ and recall is $\frac{tp}{tp+fn}$. These measures are unable to individually describe the system, so the F-measure is used to 'balance' them. Usually as one of them increases, the other one decreases.

Precision is 'how many instances were classified correctly' - did it misclassify many negatives as positives? Precision is the probability that a random instance that is categorized as positive is indeed positive.

Recall is how 'robust' your system is - does it miss a lot of positives and classified them as negatives? Recall as the probability that a random instance that is actually positive is correctly categorized as positive by the system.