The Implementation of chronotonic similarity within an applet

Ludger Hofmann-Engl

Development - Croydon Family Groups

hofmann-engl@chameleongroup.org.uk

Abstract

The term chronotonic similarity/distance was first introduced by Hofmann-Engl in the late 1990ths in order to describe the phenomenon of what might be called rhythmic similarity, and although it has been used within a number of studies and in particular in order to classify flamenco meters, it appears that no attempt has been undertaken so far to produce an application which would allow potential users to compute the chronotonic similarity via a user friendly interface. This paper introduces the reader to the concept of chronotonic similarity and discusses the method which was used to implement this similarity measure within an applet. The paper also touches on issues of ongoing development and the integration of the software into other applications such as automated composers, automated analysis and search engines.

1. Introduction

During the 90ths, Hofmann-Engl realized that rhythmic, or to use his terminology, chronotonic similarity could only be computed if rhythms were to be represented in atomic notation in analogy to a concept introduced by Gustafson (1987, 1988) in the context of phonetics. In more recent years, the concept of chronotonic similarity or chronotonic distance has been successfully used in the attempt to classify flamenco music (compare J. M. Diáz-Báñez, G. Farigu, F. Gómez, D. Rappaport & G. T. Toussaint, 2004). In a very recent study by Thul & Toussaint (2008), the chronotonic distance was successfully used to analyse African Timeliens and North Indian Talas. However, due to some mathematical difficulties and the need of computer aided computation has hindered a more comprehensive application and evaluation of chronotonic similarity and its development. Therefore, the author decided to program an applet which allows for the effective and user friendly computation of chronotonic similarity. The applet is located at:

www.chameleongroup.org.uk/software/c-chain.html

This paper will explain the concept of atomic notation and the chronotonic similarity based upon it. It then will strive to explain how the chronotonic similarity has been implemented within a java program and will conclude with a brief outlook on potential future developments and implementations.

2. Atomic Notation

In order to introduce atomic notation, two methods are available. Firstly, a formal mathematical introduction could be given and secondly a more practical and example-based dissemination. The author decided to adopt the latter method.

Let us consider the two following rhythms:

rhythm 1: 1/4 ½ 1/8 1/8 ½

and

rhythm 2: ½ 1/16 1/16 1/8 1/4 ½

Now, the smallest common element in both rhythms (c-chains) is the sixteenths value in as much as it goes into the 1/8 twice, the 1/4 four times, the ½ eight times. Thus, the ½ note is 8*8 or [8, 8, 8, 8, 8, 8, 8, 8], the 1/4 is 4*4 or [4, 4, 4, 4], the 1/8 is 2*2 or [2, 2] and the 1/16 is 1*1 or [1]. Representing the rhythms this way, we can write:

rhythm 1: [4, 4, 4, 4; 8, 8, 8, 8, 8, 8 ,8, 8; 2, 2; 2, 2; 8, 8, 8, 8, 8, 8, 8, 8](1/16)

and

rhythm 2: [8, 8, 8, 8, 8, 8, 8, 8; 1; 1; 2, 2; 4, 4, 4, 4; 8, 8, 8, 8, 8, 8, 8, 8](1/16)

Further details of this form of notation can be found in particular in Hofmann-Engl (2003a). However, for the purpose of this paper, the author believes that the above example might be sufficient for now.

3. C-chains and chronotonic distance

Instead of speaking of rhythms, we will from now on speak of c-chains which stands for *chronotonic* chains. The reason is as pointed out by Hofmann-Engl (2003a), that the meaning of rhythm is too loaded and ambivalent.

Returning to our example above we are dealing with two c-chains c_1 and c_2 with:

c_1 = [4, 4, 4, 4; 8, 8, 8, 8, 8, 8 ,8, 8; 2, 2; 2, 2; 8, 8, 8, 8, 8, 8, 8, 8](1/16)

and

c_2 = [8, 8, 8, 8, 8, 8, 8, 8; 1; 1; 2, 2; 4, 4, 4, 4; 8, 8, 8, 8, 8, 8, 8, 8](1/16)

Plotting both chains within a graph, we obtain:

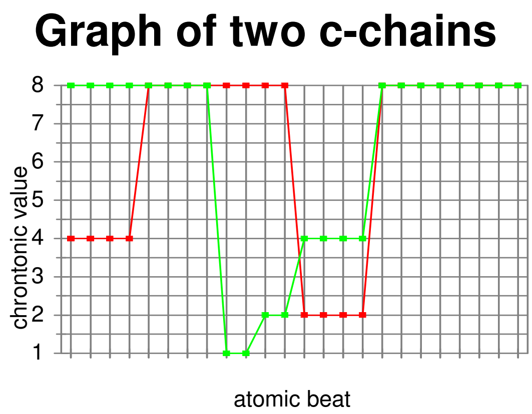

Figure 1: Illustrating the plot of 2 c-chains

The chronotonic distance can be simply defined as the distance between all chronotonic values of the two chains along all atomic beats. In our case we obtain the overall distance:

D = 4 + 4 + 4 + 4 + 0 + 0 + 0 + 0 + 7 + 7 + 6 + 6 + 2 + 2 + 2 + 2 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 = 50

This make an average distance of 50/24 = 2.08.

4. Chronotonic similarity

The chronotonic similarity is correlated to the chronotonic distance and a tempo adjustment. That is, two identical rhythms played at different speed become increasingly dissimilar with increasing tempo difference.

According to Hofmann-Engl's (2003b) experiment on chronotonic similarity, the tempo difference can be captured by the following formula:

$$F_3 = e^{-k_3 \ln^2 a} \quad (1)$$

with F_3 as the function predicting the similarity according to tempo differences, k_3 as a constant and *a* the tempo difference between the two c-chains.

We give an example:

c_1 = ½ ½ 1/4 1/4 and c_2 = 1/4 1/4 1/8 1/8

Both chains have the same chronotonic shape, but c_2 has half the tempo of c_1. Substituting k_3 by the value 0.15 (which delivered the highest correlation within Hofmann-Engl's experiment (2003b)), we compute:

$$F_3 = e^{-0.15 * \ln^2 1/2} = 0.95$$

The chronotonic distance is a second factor and computed according to Hofmann-Engl (2003b) as:

$$\left\| \vec{F}_1 \right\| = \sqrt{\frac{\sum_{i=1}^{n} \left(e^{-\frac{k_1}{n} s_i^2} \right)}{n}} \qquad (2)$$

with $\left\| \vec{F}_1 \right\|$ as the chronotonic distance measure, *n* the amount of atomic beats, k_1 as a constant and s_i the chronotonic distance at the place *i*.

We give an example:

c_1 = ½ ½ 1/4 1/4 and c_2 = 1/8 1/8 1/4 1/4

The length of c_1 is 1.5 beats and the length of c_2 is 0.75 beats. In order to compute the chronotonic distance similarity, we either have to stretch c_2 by a factor 2 or shrink c_1 by half. We stretch c_2 by the factor 2 and obtain:

c_2' = 1/4 1/4 ½ ½

with c_1-chain = [2, 2; 1; 1](1/4) and
c_2'-chain = [1; 1; 2, 2; 2, 2]](1/4)

Setting k_1 as 1.28 , we obtain:

$$\left\| \vec{F}_1 \right\| = \sqrt{\frac{\sum_{i=1}^{n} \left(e^{-\frac{1.28}{n} * s_i^2} \right)}{n}}$$

Computing the tempo similarity and the chronotonic distance similarity, we obtain the values $F_3 = 0.95$ and $\left\| \vec{F}_1 \right\| = 0.78$. Setting the overall similarity as $S = F_3 * \left\| \vec{F}_1 \right\|$, we obtain the chronotonic similarity $S = 0.74$.

5. The applet

The applet has been uploaded to:

www.chameleongroup.org.uk/software/c-chain.html

and includes the source code. The GUI is depicted in the picture below.

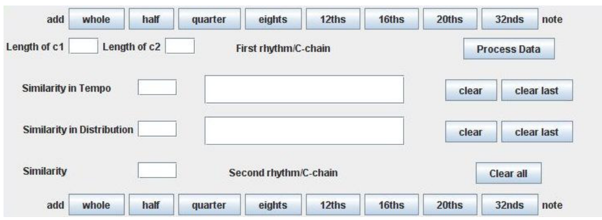

Figure 2 depicting the GUI of the chronotonic similarity applet

The top row allows for inputting the chronotonic values for the first chain and the button row for the second chain. The values which can be inputted are whole notes, half, quarter notes etc. The two windows in the middle display the chains as they are inputted. Once the data are being processed by pressing the *process data* button, the length of both chains are displayed in the windows at the top left, the window below displays the tempo similarity followed by the chronotonic distance similarity and below this the overall chronotonic similarity is shown. The buttons on the right are correct and clear buttons.

However, the more interesting question is to do with the issue of how the algorithm has been implemented.

6. Processing tempo similarity

In a sense the program does not process real time values, but relative values only. This means that making use of this program requires to derive the relative values of the two c-chains (rhythms) to be compared. For instance, if one rhythm has the tempo marking ♩ = 120 and the other rhythm the marking ♩ = 60, quarter notes of the first c-values are to be inputted as half note values or the quarter notes of the second chain as eighths notes.

The program computes the tempo similarity by adding all c-values for the first chain and adding all c-values for the second chain. The tempo similarity value then is computed according to formula (1). The relevant passage in the program is shown below:

```
for (int k = 0; k < i; k++)

{
SumV1 = SumV1 + v1[k];
}

for (int k = 0; k < j; k++)

{
SumV2 = SumV2 + v2[k];
}
v = 0.15 * Math.pow(Math.log(SumV1/SumV2),2);
F3 = Math.pow(2.17, -v);
F3 = Math.round(F3*10000);
F3 = F3/10000;
SumV1 = Math.round(SumV1*1000);
SumV1 = SumV1/1000;
SumV2 = Math.round(SumV2*1000);
SumV2 = SumV2/1000;
```

7. Processing the chronotonic distance similarity

Processing the chronotonic distance similarity does not follow the method described above. However, the first step does follow the method by bringing the two c-chains to equal length and in a first instance, the c-chains need to be brought to equal length and the program reads:

```
if (SumV2 > SumV1)

{
L = SumV2/SumV1;

for (int m = 0; m < i; m++)

{
v1[m] = L*v1[m];
}
}

if (SumV1 > SumV2)

{
L = SumV1/SumV2;

for (int m = 0; m < j; m++)

{
v2[m] = L*v2[m];
}
}
```

However, the next step does not follow the method of finding an atomic beat and representing the c-chains in atomic notation. Instead the chains are constructed so, that in intervals of 0.005, the c-values for both chains are computed. The relevant passage for the first chain reads:

```
for (int m = 0; m < i; m++)

{
for (double k= 0.005; k < v1[m] ; k+=0.005)

{
al1++;
a1[al1] = v1[m];
}
}
```

This produces two quasi-atomic chains. Note, for a more precise computation it would be sensible to set k+=0.00005.

The difference between the two chains is in analogy to Hofmann-Engl (2003b), set to be the logarithmic difference, and the program reads:

```
for (int m = 1; m < al1; m++)

{
T[m] = Math.log(a1[m]/a2[m])/Math.log(2);
}
```

As we are dealing now with a large number of *m* (corresponding to the number *n* in formula (2)), it became obvious that *n* had to be taken out of the original formula and so we write:

$$\left\|\vec{F}_1\right\| = \sqrt{\frac{\sum_{i=1}^{n}\left(e^{-k_1 s_i^2}\right)}{n}}$$

where T[m] = s_i^2

Additionally, the value of k_1 seemed too small, and so it was replaced by the value k_1 = 12.8. However, further experimentation is necessary in order to determine more precise parameter fixing.

The relevant passage reads:

```
for (int m = 1; m <al1; m++)

{
o[m] = Math.pow(T[m],2);
p[m] = Math.pow(2.17,(-12.8*o[m]));
T1 = T1 + Math.pow(p[m],2);
}

T1 = Math.sqrt(T1/al1);
T1 = Math.round(T1*1000+1);
T1 = T1/1000;
```

8. Conclusion

As far as the author is aware this is the first attempt to produce a program which computes the chronotonic similarity.

The GUI seems user friendly but will need more time values and perhaps data entry via a TexField would be beneficial.

Other issues relate to the clear buttons which seem not to work reliable once a computation has been executed. Further, more experiments are required in order to fix the parameters more precisely. However, the results appear intuitively sensible.

It appears likely that the program could not only be useful for analytical purposes but could be integrated within musical search engines. Finally, the software could become part of an automated composer by producing random variations within a given similarity level. Finally, the classification of ethnomusicological material can profit from the implementation of chronotonic similarity/distance measures (e.g. Thul & Toussaint, 2008).

References

Diáz-Báñez, J. M., Farigu, G., Gómez, F., Rappaport, D. & Toussaint, G. T. (2004). El Compás Flamenco: A Phylogenetic Analysis. In: Proceedings of BRIDGES: Mathematical Connections in Art, Music and Science, Southwestern College, Winfield, Kansas, pp. 61-70.

Gustafson, K.(1998). The graphical representation of rhythm. In (PROPH) Progress Reports from Oxford Phonetics, vol. 3, pages 6 - 26, University of Oxford.

Gustafson, K. (1987). A new method for displaying speech rhythm, with illustrations from some Nordic languages. In K. Gregersen and H. Basbøll, editors, Nordic

Prosody IV, pages 105 - 114, Odense University Press.

Hofmann-Engl, L. (2003a). Atomic notation and melodic similarity. In: Proceedings of Computer Music Modelling and Retrieval (CMMR 03), Montpellier, France

Hofmann-Engl, L. (2003b). . Melodic Similarity and Transformations: A theoretical and empirical approach. PhD thessis, Keele University

Thul, E. & Toussaint, G. (2008). A Comparative Phylogenetic Analysis of African Timelines and North Indian *Talas*. In Proceedings of BRIDGES-2008