Internship Report

Live algorithms with complexity matching

Author :
Guillaume Robal

Supervisor :
Tim Blackwell

Goldsmiths Computing Team
University of London,
New Cross, London, SE14 6NW, UK

August 11^{th}, 2014

Acknowledgments

I would like to thank mister Robert Zimmer and mister Tim Blackwell who have accepted me on Goldsmiths campus for my internship in the computing team. I particularly thank mister Blackwell for his supervision of my assessment and for granting me a few leaves to fly back my home and prepare an important personal event.

I would like to thank the ATIAM supervision team and the IRCAM for this great year between these walls among fantastic people.

I would also like to thank my family for their help and support during the difficult period spent in trying to settle in London for the last few months.

Last but not least, I want to thank my future wife without whom nothing would have been possible.

Summary

The goal of this assessment was the study of a new parameter issued from the information theory within the field of music information retrieval and the development of a software letting the machine improvise from an input audio signal whose source remains undetermined (noise, musician, other). We had to get the signal from the micro input and make it pass through information technology methods developed to obtain a complexity characterising the sound object from simple to complex.

From a set of complexity measures and their evolutions, we had to develop methods matching the complexity value of the signal played by the machine. Ideally, we would have liked to get a button able to modify at will the output complexity by managing the different sounds parameters involved.Therefore, we have developed a synthesizer, a digital appliance able to reproduce distinct sounds from given parameters.

Keywords : Information theory, complexity, excess entropy, granular synthesis

Résumé

L'objectif de ce stage était l'étude d'un nouveau paramètre issu de l'informatique théorique dans le cadre de la recherche en informatique musicale et le développement d'un logiciel permettant à la machine d'improviser à partir d'un signal audio dont la source reste indéterminée (bruit, musicien ou autre). Nous devions capter le signal à l'entrée du micro et le faire passer au travers de méthodes informatiques développées afin d'en obtenir une complexité caractérisant l'objet sonore mesuré de simple à complexe.

A partir d'un ensemble de ces mesures de complexités et de leurs évolutions, nous devions développer des méthodes décidant la valeur de la complexité du signal joué par la machine. Idéalement, nous aurions voulu obtenir un bouton capable de modifier à volonté la complexité de sortie en gérant les differents paramètres sonores mis en jeu. Nous avons pour cela développé un synthétiseur, objet numérique capable de reproduire à partir de paramètres donnés un ensemble de sons distincts.

Mots-clés : Théorie de l'information, complexité, entropie d'excès, synthèse granulaire

Contents

Introduction

This work deals with the notion of complexity in music. Before speaking of music, we need to define what the word complexity means. If you look it up in a dictionary, you will probably find the following definitions: "The quality or condition of being complex" and "the state or quality of being intricate or complex"[1]. Edgar Morin[2] says about complexity: the word complexity is in itself a problem word, the matter is that our knowledge doesn't allow us to clear it up. Saying "this is complex" implies a puzzle to our understanding, indicating an inability to usual straightforward comprehension [Morin, 1977].

The etymology of the word complexity comes from the fourteenth century Latin "complexus" compound of 'com' (together) and 'plecto' (to weave, to braid) meaning to interlace an aggregate of parts. The word complexity is used in many fields today from biology to chaos theories in natural sciences but also in social sciences from the beginning of the twentieth century. The natural sciences cartesianism paradigm from the seventeenth century to the nineteenth century was founded on the Platonician thought (finding the harmonious orders of the spheres) of rationalism (the knowledge of the natural principles obtained by the mind is supposed immutable) and reductionism. The ideal was to find simplicity in nature.

In the nineteenth century some major discoveries in natural sciences point out the importance of new features: uncertainty and emergence. In fact, some regularities are discovered in disorder (Boltzmann equations) and also a randomness selection in the evolution of species (Darwin). In the twentieth century we witness a breach from the Platonician thought and the beginning of complexity sciences: Henri Poincaré, a French mathematician, discovered that the problem of the three bodies (celesta mechanic) has no analytical solution (without infinite series), the state of the bodies depending on initial conditions[3]. In the 1930's, the incompleteness theorem expounded in Alan Turing's and Kurt Gödel's work shows that there is no coherent and complete axiomatic system for arithmetic, a result which creates a trouble in the mathematician community: there exist

[1] Definitions obtained from http://www.thefreedictionary.com/complexity

[2] Edgar Morin is a French sociologist and philosopher specialised in the notion of complexity, emeritus research director and president of the scientific CNRS (french national scientific research center) counsel, he is also the president of the association for the complex thought and deals with the transdisciplinarities of sciences concerning the notion of complexity.

[3] Henri Poincaré opened through this way what will be the chaos theory field. However biased he was, due to the period he lived in, he was also criticizing the study of minor random events: "the facts that are the most useful are the fact which may be repeated many times" and "there is a hierarchy of facts, some are without sense; they do not convey anything else but what is inherent to themselves. The scientist who has observed them has learnt nothing but one fact, and is unable to predict any new facts" (translated from french online book, "Oeuvres", Henri Poincaré).

undecidable theorems, which are undemonstrable and their negations are also undemonstrable. In the 1960's, Gregory Chaitin worked on the algorithmic information theory and showed that some questions in the number theory have random answers. His work is based on the information theory developed by Shannon and Weaver in 1948. Their fundamental work which we will develop in the next chapter had a fundamental effect on complexity sciences. After 1950 and for the first time in human history, there are more scientific articles dealing with complexity than with simplicity.

We have just spoken about complexity from the mathematics viewpoint but we could have developed it in the same way in biology, physics and so on. The point was to explain the emergence of a word which seems to be used in many fields without apparent links between one and other which are in fact related by a change of paradigm thought through centuries. We must also explain that this notion has been thought by philosophers throughout history such as Gaston Bachelard who has also been involved in this paradigm change. Gaston Bachelard in 1934 said "Science isn't studying a simple universe, but a complex universe which it denatures by simplifying it, then confounding it with reality." For him, like for Aristotle, "the whole isn't the sum of the parts". To simplify it, in natural sciences, complexity deals with the chaotic nature or tendency measure of a system but also the emergence of new behaviours. But what about its place in music?

Music complexity is often described in art terms. Searching for computational complexity in research papers, we could only get few results using entropy measures and Kolmogorov complexity. We believe that our approach might bring some new knowledge to this field with the notion of excess entropy. An introduction to general science and computational complexity might be read in [Mitchell, 2011] and some information about music complexity in [Streich, 2006]. The purpose of Tim Blackwell's live algorithms is to define a set of music complexities through time not based on abstract cultural parameters but on the organisation of a set of minimal behaviours ruling musical attitudes.

Here we have tried to develop algorithms able to match a complexity measure and to decide a complexity output in real time. It was difficult for us to choose parameters which were able to compute in very little time to make decisions without interrupting the output stream and for the stream to be able to evolve consequently to input.

Live Algorithms are software meant to calculate in real time. They can consist in only an analysis in real time or a synthesis with parameters defined by a machine or a human being. Tim Blackwell has developed the P-F-Q modular architecture: P standing for analysis, Q for synthesis and f for the artificial intelligent part deciding the pattern to play. In the next chapters, we will study the behaviours of P and Q. We will then present some ideas on f and finally we will show the results we got.

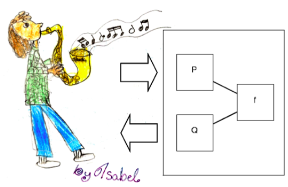

Figure 1: Diagram of a live algorithm showing analysis (P), synthesis (Q) and patterning (f) modules (from Tim Blackwell's website)

Chapter 1

Analysis

1.1 Generalities

In this chapter, we are going to see how to estimate complexity measures through information theory tools. This work is based on the information theory developed by Shannon and Weaver in 1948. Shannon used to work as an electronics engineer in telecommunications at Bell Laboratories and intended to develop a cheaper way to send messages and a method to reduce mistakes on message transmissions. Messages are encoded to become shorter and then converted into electric signals (waves or binary data) and sent through channels before being received. As any channel provokes noise, some errors could be send and perturbed the message. Shannon wanted to know how much information was needed to send to make sure that the message was transmitted correctly.

This principle is quite simple: imagine you have a message to send over using an alphabet $\mathcal{A}$ of four letters {A,B,C, D}:

Example 1 A A A A B A A C A A D A C A C A A A B A

The message is of size twenty, there are 14 A, 2 B, 3 C, 1 D. So, you have 70 percent of A, 10 percent of B, 15 percent of C and 5 percent of D. If you record the values of the alphabet by a growing appearance chance, you get D-B-C-A.

Let's suppose you intend to transmit these values with minimum information, we use a method called "divide to rule". It is quite the same trick you will be used if you want to discover a number between 1 and 100. If you start by choosing the right value randomly, let's says twenty, then thirty if I tell you it's more, you can easily imagine it could take you many steps to guess. If you start by choosing the middle value like 50 in the first case you will gain more time because you are eliminating half of the values each time. Imagine my value is 100, then you will probably choose 50, 75, 88, 94, 97, 99, 100 so seven values in an extreme case to guess with this method. Mathematically, we can also see that $2^7 = 128$, so if you are dividing into two, you should find any value between 1 and 128 in seven steps.

In example 1, we similarly separate the four letters by the size of the maximum percentage. We could separate them in four groups giving each letter same weight but there

would be no interest as there would be no economy.
As A already represents 70 percent of the message and so more than half its weight, we get two groups D-B-C (30 percent) and A. Indeed, if we want the message to take as little space as possible, we will give the most frequent figure the lowest value. So let's assume giving 0 to the group D-B-C to begin with and A with 1. Then let's do it again with the subgroup D-B-C, we will get D-B equals 15 percent and C also equals 15 percent so D-B will start with 00 and C with 01. There is a final step for the last subgroup D-B with 5 percent for D and 10 for B so we have 000 for D and 001 for B. This method has the particularity of giving an efficient method to encode economically the message, each A composing 70 percent of the message being encoded by one value (1), C by two (01) and D and B by three values (000 and 001).

To make the matter more explicit and show how economic it is, if we had given each value A, B, C and D the same weight, let's say (00,01,10,11) to encode them, each value coded by two numbers, we would have needed 40 symbols to send this message of size twenty. With our method 14 A of size one plus 2 B of size two plus 1 C of size three plus 1 D of size three, we only need 24 symbols to encode the message. This is known as Shannon-Fano algorithm. A tree corresponding to the previous results is shown in figure 1.1.

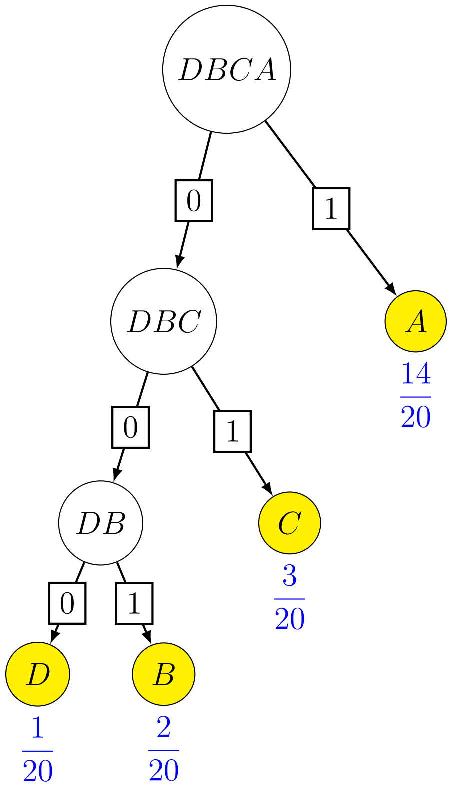

Figure 1.1: Shannon-Fano algorithm applied to example 1

As we sum up now, the Shannon and Weaver communication model proposes a coding source and decoding receipt after a passage through a channel.

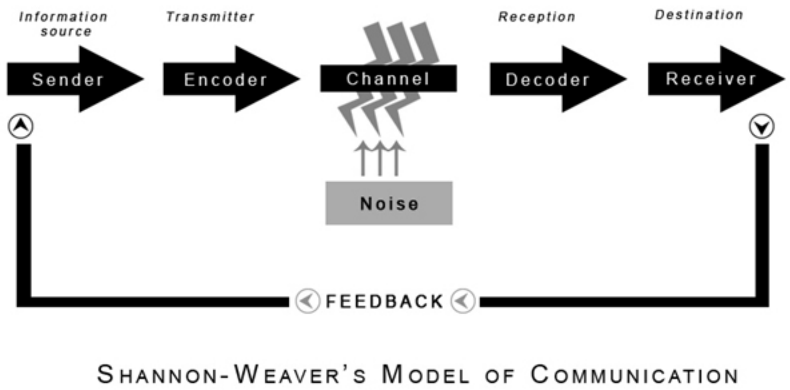

Figure 1.2: Shannon-Weaver's communication model

A model of encoding and decoding source may be a phone with waves or a computer with binary digits. The main trouble is as you can see in figure 1.2 that due to physics limits, we can't create a channel without noise. This noise creates errors which may modify the value of the data transmitted. Example 1 is starting with "A A A" which is encoded "1 1 1". If the second number transmitted is due to an error converted to a zero you have at the decoding step "1 0 1" corresponding in figure 1.1 to the message "A C". The feedback is here to ensure that the message transmitted corresponds to the message sent.

Let's define things a little more formally. As we don't usually speak in terms of percentage but in terms of probability, let's define probability: it corresponds to the presence rate of a symbol when an infinity of symbols are present.

$$P(S = s_i) = \lim_{N->+\infty} \frac{N_i}{N} \tag{1.1}$$

with S representing all possible symbols and $N_i = \#s_i$, number of times s_i symbol appears.

In example 1, we only have an approximation of the probability as the size of the message is finished but we will not consider this difference. We write $P(S = A) = 0.7, P(S = B) = 0.1, P(S = C) = 0.15$ and $P(S = D) = 0.05$.

In this section, we have used examples to try and explain in a very simple manner or recall to mind how information theory works. In the next section, we will continue and provide definitions and applications to the analysis system of our software.

1.2 Definitions and applications

As we are using a "divide to rule" law in this field, we need to use the logarithm of base two. We already used it to find the correct number, we needed seven steps as $log_2(128) = 7$.

$$2^x = y \Leftrightarrow log_2(y) = x \tag{1.2}$$

The base 2 logarithm is particularly adapted to binary signals as used in computers. As we will only consider this case we will speak about bits of information. In information theory, bits do not represent the "binary digits" but the degree of minimal information transmitted in a message[1]. If we want to define a surprise effect, we will say that a surprise event comes when the probability of this event is weak: $p = \frac{1}{2^s}$. From this result we measure the surprise as :

$$s = log_2(\frac{1}{p}) = -log_2 p \tag{1.3}$$

We can easily extend this notation to each symbol by $s_i = -log_2 p_i$. If we want now to measure the average surprise effect, we have to add up the surprise over all symbols with the probability of each of them. We get the famous entropy formula:

$$H[X] = -\sum_{i=1}^{\#symbols} p_i log_2 p_i \tag{1.4}$$

In information theory vocabulary, X is the source and H(X) is defined as the measure of the uncertainty of any system studied. As written in the introduction, the disorder regularities were observed for the first time with Boltzmann's equations. Boltzmann defined entropy constant as :

$$H = -k \sum_{i=1}^{\#states} W_i log_2 W_i \tag{1.5}$$

where W_i is the thermodynamic probability of each state, k is Boltzmann's constant [Loy, 2006]. It was not even Shannon but mathematician Von Neumann who gave to formula (1.4) the name of entropy after observing similarities in both equations.

In our tests, we are working with real data. We hope to get some information relative to expectations and complexity with analysis of incoming input signals. In example 1, we were dealing with a set of symbols in a message. The plain results were due to some redundancy in the message with few different symbols. The main trouble with real signals is that they have an important number of different amplitudes through time or in each spectrum of the Fourier transform (spectral image of a time signal window).

If we are trying to be extremely precise, we will have to define an important alphabet and then an important number of symbols to treat with few possibilities to compare them. We are starting our algorithms with a step of quantization. In this chapter, we will just

[1] In information theory, it is quite common to obtain non integer bits of information.

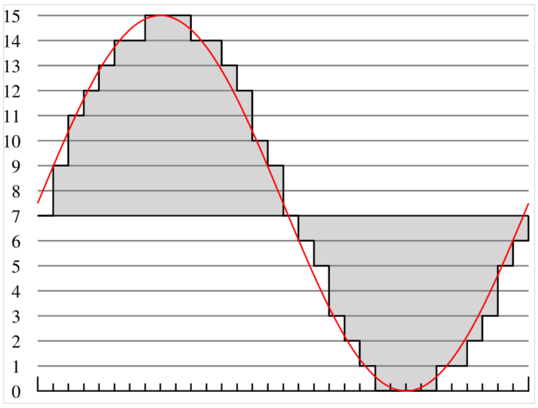

Figure 1.3: An example of quantization 2^N amplitudes kept ($N = 4$) as a function of time

explain the general method, then we will explore the different methods tested in another chapter.

A computer is always going through a method of quantization to set amplitudes because it cannot store an infinite number of values. It generally stores them in a number of values in a power of two, due to the number of bits affected, (sometimes $2^{16} = 65536$ amplitudes). If we get too much precision in the quantization, two close amplitudes could be taken as different. At the same time, we cannot take an infinite number of bins through time or for spectrum. Generally, we have an upper value in the time domain with most of the times 44100 samples per second, which means we could have very few repetitions in the same second with an important number of amplitude bits.

If we want to get some estimation of the entropy of the signal, we observe that we need to refine this step of quantization and also choose an adapted time or spectrum windowing to get the information needed by matching each quantized amplitude to a symbol. The figure 1.3 represents an example of quantization.

In this assessment, we are also studying two complexity measures : Kolmogorov complexity measure and excess entropy measure.

The Kolmogorov complexity measure is a theoretic measure which calculates the minimal numbers of characters necessary to build a software. This absolute definition is purely theoretical because the evaluation of the minimality of the software would give a greater number of characters than the software itself. Then, we can only get an upper boundary of the Kolmogorov complexity. We will estimate it from Lempel-Ziv algorithm.

The Lempel-Ziv algorithm goal is to obtain a dictionary of symbols and associate a probability to each symbol. The dictionary may be seen as $\mathcal{A}$ in the example 1. For each window analysed, if a new symbol is created, the size of the dictionary grows up and we associate a value of one to a counter along the new "definition" in the dictionary which we will qualify as address to keep a computing vocabulary. Otherwise, whenever a symbol already occurring in the dictionary is called we increment the counter associate to the address of the symbol. An upper boundary of the Kolmogorov complexity may be estimated as the number of values in the dictionary: the more randomly the signals, the more addresses in the dictionary and so the higher the complexity. When the time of analysis comes, we create a copy of all the values associated to each symbol in a new tab and divide them into the number of symbols according to formula (1.1). We then obtain a probability associated to each symbol.

We may now apply entropy formula (1.4) but as we get the quantized signal, we might also want to know the number of times a signal has the same evolution, we can do it easily by adapting a dictionary for a number of larger size symbols which are inferior or equal to expressible L. The dictionary is now seen as $\mathcal{A}^L$.

$$\overleftrightarrow{S} = ...s_{-2}s_{-1}s_0s_1s_2.... \tag{1.6}$$

In formula (1.6), $\overleftrightarrow{S}$ represents the theoretic infinite chain of symbols. If we take L symbols each time and shift from one symbol at a time, we can then define block symbols as formula (1.7) and total entropy as formula (1.8).

$$s_i^L = s_i s_{i+1} ... s_{i+L} \tag{1.7}$$

$$H(L) = -\sum_{s^L \in \mathcal{A}^L} p(s^L) log_2 p(s^L) \tag{1.8}$$

In formula (1.8), the sum is understood to run over all possible blocks of L consecutive symbols. However, as L is supposed strictly positive, we define $H(0) = 0$ [Crutchfield and Feldman, 2003].

We can now see our dictionary as a set of sub-blocks of size $\{1, 2..., L-1, L\}$, each of the sub-block containing a set of symbols of the size of the sub-block and Kolmogorov complexity as the number of different symbols of size less or equal than L expressible.

If we want to be able to calculate a complexity measure, we need to introduce two new features, the source entropy rate and the excess entropy. The source entropy h_μ is the rate of increase with respect to L, it represents the irreducible randomness or unpredictability in sequences produced by a source[2] [Crutchfield and Feldman, 2003] [Crutchfield, 2003].

$$h_\mu = \lim_{L->+\infty} \frac{H(L)}{L} \tag{1.9}$$

However, contrary to formula (1.9), we cannot work with infinite sequences as we only have finitary processes. We then define $h_\mu(L)$ as the estimate of the source randomness for only blocks of symbols of size up to L [Crutchfield and Feldman, 2003]. The source entropy rate may be approximated by the source entropy gain $\Delta H(L)$ which represents the pseudo derivative of $H(L)$.

[2]The unit are in bit/symbol.

$$h_\mu(L) = \Delta H(L) = H(L) - H(L-1). \tag{1.10}$$

The equation (1.9) also gives an idea of the behaviour of H(L) when L tends to infinity, we can then define equation (1.11).

$$H(L) \sim h_\mu L \text{ as } L \to \infty \tag{1.11}$$

As mentioned before, h_μ is supposed to represent the irreducible unpredictability of the source so if we take the difference between the entropy of the source and h_μ for each L, we will obtain redundancy for each symbol in equation (1.12).

$$R(L) = H(L) - h_\mu L \tag{1.12}$$

We have with equation (1.12) a redundancy of the source per blocks of symbols of size L but another approach is to consider a per-symbol L redundancy which measures the per-symbol of length L entropy rate estimate excess to the per-symbol entropy. It is expressed as the derivative of R(L) and is referred to as the local predictability [Crutchfield and Feldman, 2003].

$$r(L) = \Delta R(L) = h_\mu(L) - h_\mu \tag{1.13}$$

Tim Blackwell's idea was to use excess entropy (E). This is a feature never used before in music information retrieval. We are defining it as the intrinsic redundancy of the source: the sum of the individual per symbol L redundancies:

$$E = \sum_{L=1}^{+\infty} r(L) \tag{1.14}$$

However, as we are using finitary processing, we use equations (1.11), (1.13) and (1.14) to define a finite L-expression for E:

$$E(L) = H(L) - Lh_\mu(L) \tag{1.15}$$

The interest of excess entropy is that measuring the redundancy per symbol of a signal gives us an idea of the complexity of this signal. In music, we may define the simplest signal as pure tone and the most complex as white noise[3]. Between both extremes, we have an infinity of more or less complex signals. However, if we consider the perception approach, the white noise bears no information in itself as no rules are predictable. By calculating the redundancy, the excess entropy of the source will give the same weight to a pure tone as it is already a completely compressed signal without redundancy in the source entropy and to the white noise since it is a decorrelated signal without redundancy possible in time domain and it has the same energy for each frequency bin which gives

[3] A white noise is a random process realisation where the spectral density is the same for all frequencies. It contains all frequencies in its signal without correlations between its values.

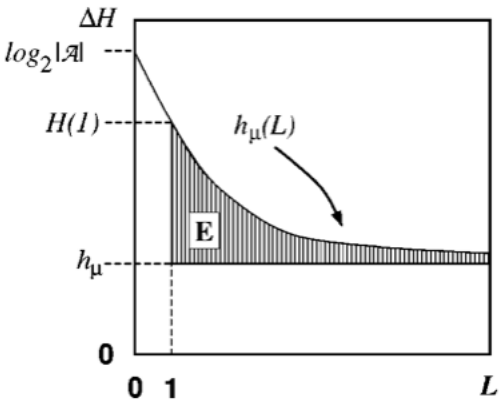

Figure 1.4: Entropy rate convergence [Crutchfield and Feldman, 2003].

a completely compressed signal in spectral domain. The more important excess entropy will then be about signals which have the most important numbers of redundancies with less compressibility possible or to express it in a simpler way, a more important evolution through time with some correlations between its values to let's us figure out patterns. You can see in figure 1.4 a reminder of the calculus of the entropy and its convergence[4].

The last theoretic element to be introduced is transient information (T). The limits of excess entropy are due to the fact that even if excess entropy provides information about the structure of the signal, it cannot observe a difference between two different signals with the same entropy (like a n-repetition of "10100" and "10010"). Transient information is meant to measure how difficult it is to synchronise to a source, to follow it. It allows us to draw structural distinctions between periodic sequences. T can also be used in a Markovian process to capture the total state-uncertainty [Crutchfield and Feldman, 2003]. It can be directly expressed by formula (1.16) but is computable only by formula (1.17), T being in this case the sum over all L as indicated in figure 1.5.

$$T = \sum_{L=1}^{+\infty} L[h_\mu(L) - h_\mu] \tag{1.16}$$

$$T(L) \equiv E + h_\mu L - H(L) = 0 \tag{1.17}$$

[4]In this figure, $|\mathcal{A}|$ designs the size of $\mathcal{A}$ and $log_2|\mathcal{A}|$ the theoretic capacity of the channel.

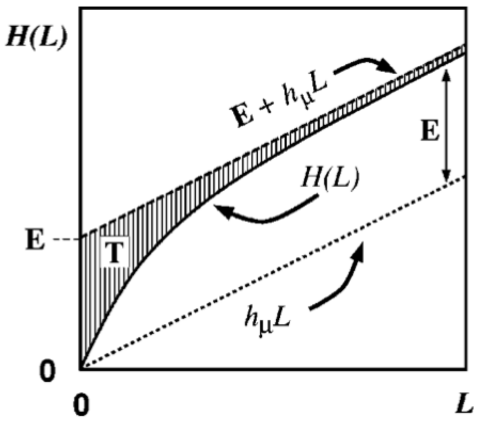

Figure 1.5: Total Shannon entropy growth for a finitary information source [Crutchfield and Feldman, 2003].

Chapter 2

Synthesis

In the previous chapter we have developed tools to use symbols and get probabilistic laws. One of the goals of this assessment was to develop a machine able to play a large panel of sounds and to develop a seemingly non-human behaviour dealing with probabilities. In order to play electronic music, we use a digital appliance called synthesizer able to reproduce a set of sounds depending on its inner structure. In this chapter, we will present the different ideas we have had to choose a kind of synthesis method and have used to get a synthesizer playing in real time.

2.1 Study approach

In the preface of [Xenakis, 1963], Iannis Xenakis expounded : "It is not so much the inevitable use of mathematics that characterizes [...] as the overriding need to consider sound and music as a vast potential reservoir in which a knowledge of the laws of thought and the structured creations of thought may find a completely new medium of materialization, i.e., of communication, For this purpose the qualification"beautiful" or "ugly" makes no sense for sound, nor for the music that derives from it; the quantity of intelligence carried by the sounds must be the true criterion of the validity of a particular music".

As one of the goals of this assessment is to build a machine able to play sounds without any human knowledge[1] but only from music complexity, we have considered Iannis Xenakis's approach as a way to produce relevant results.

Iannis Xenakis was a composer and a mathematician who wanted to use a probabilistic approach in his work. In 1955, he criticised the limits of polyphony and serialism : "Linear polyphony destroys itself by its very complexity; what one hears is in reality nothing but a mass of notes in various registers. The enormous complexity prevents the audience from following the intertwining of the lines and has as its macroscopic effect an irrational

[1]In his work, Tim Blackwell develops the idea that an independent machine could teach us more than one using a deterministic behaviour. We try to develop a machine able to play with humans as an extraterrestrial discovering human music, expecting to get emerging patterns from it. However, we believe we can't expect full emergences with respect to humans tonality rules playing.

and fortuitous dispersion of sounds over the whole extent of the sonic spectrum. There is consequently a contradiction between the polyphonic linear system and the heard result, which is surface or mass" [Xenakis, 1955].

Gareth Loy declared in [Loy, 2006] (p.333) that Iannis Xenakis attempted to align music aesthetics with a natural theory in legacy of Renaissance's composer and music theorist Gioseffo Zarlino who discovered that singers could use much more precise sounds than the defined twelve notes musical temperament but rather intervals in their "true form" (in "Istitutioni Harmoniche"). Gioseffo Zarlino described a natural scale based on the most possible number of pure chords[2]. His work was then studied by many physicists such as Hermann von Helmoltz and was called Zarlino scale or natural scale. Their goal was to get out of the predefined way of equal temperament and to get the full spectrum of possible existing sounds.

From the deadlock of serialism, Xenakis founded stochastic music, a non-deterministic, random music based on probabilities. Xenakis chose this specific term because the Greek etymology of "stochos" is "aim". Randomness in music tends to reach some purpose for Xenakis. He used what we will call in the next part a granular synthesizer in some of his works (Analogique A-B for string orchestra and tape (1959)) consisting of evolutions of micro-sounds through all spectrum and time.

2.2 Granular synthesis

In 1946, Dennis Gabor, future Nobel-prize physicist developed the granular synthesis. Before World War II, Dennis Gabor worked in Berlin like his colleague Albert Einstein. As Albert Einstein worked on light properties and discovered that it was made of tiny particles called photons, Dennis Gabor developed the same ideas for sounds, describing atomic particles of sounds as phonons or grains as it is known now.
Two years before Claude Shannon's Theory of communication, Dennis Gabor tried to catch local information from electric signals that the standard Fourier transform was unable to catch.

Joseph Fourier was a nineteenth century mathematician and physicist who was working on the heat phenomenon. He discovered that with periodic heating, the movement of heat was quickly becoming sinusoidal. He developed the idea that all trigonometric functions were the elementary constitutes of any periodic function and to any waveform. Huygens, a seventeenth century mathematician had already demonstrated that in music the timbre was constituted by a superposition of periodic waves. The Fourier transform is used to get the amplitude and frequency of each periodic wave of signals and the reverse Fourier transform to rebuild the time signal from its frequencies.
We get from Fourier's works that any function f in $[0, 1)$ could be represented by trigono-

[2] Gioseffo Zarlino works were in a period of time when the christian Church started to accept new intervals forbidden before because considered as dissonant and non representative of "pure beauty". As time made them acceptable and the history of western music has been built on the acceptance of new sounds and hearing disposal through time, Xenakis who lived many centuries later, after notably the first quarter of twentieth century bruitist period, thought that it was time to accept all intervals.

metric series[3]:

$$f(t) = \sum_{n\in\mathbb{Z}} \hat{f}(n)e^{2\pi int} \tag{2.1}$$

where

$$\hat{f}(n) = \int_0^1 f(t)e^{-2\pi int}dt \tag{2.2}$$

Fourier transform is very useful to get some values from a temporal domain but it has its own limits not only for computing[4] but also for analysing signal local time evolution. In fact, the module of the Fourier transform gives us the spectrum but as $|e^{2\pi int}| = 1$ for all n and t, it can't capture at all local perturbations[5].
This result is quite logical knowing Heisenberg's uncertainty principle (cf. equation (2.5)) explaining you cannot be as precise in time and in frequencies. However, as the Fourier transform sums all times to get frequencies and its reverse sums all frequencies to get times, Dennis Gabor wanted a middle way to get both dimensions.

$$S(k) = \sum_{n=0}^{N-1} s(n)e^{-2\pi ikn/N} \tag{2.3}$$

$$s(n) = \frac{1}{N}\sum_{k=0}^{N-1} S(k)e^{2\pi ikn/N} \tag{2.4}$$

$$\Delta f \Delta t \geq A \tag{2.5}$$

where A is represented by a quite small number.
Dennis Gabor expanded f to be translated through time and frequencies (modulation). He wrote

$$f(t) = \sum_{n,m\in\mathbb{Z}} c_{m,n}g_{m,n}(t) \tag{2.6}$$

where $c_{m,n}$ are time-frequencies mean center of$(\Delta m, \Delta n)$ (frequency m as a function of time n) (cf. figure 2.1) representing one quantum of information and $g_{m,n}$ are given by:

$$g_{m,n} = g(t-na)e^{2\pi imbt},\ m,n \in \mathbb{Z} \tag{2.7}$$

with {a,b} time-frequency shift parameters ({a,b} > 0) [Hans G. Feichtinger, 1998].

[3] $\hat{f}$ means the Fourier transform of f.

[4] In fact, as any signal must be stored in a finite number of time values, a discrete version of equation (2.2) must be applied in computer science (cf. equation (2.3)) and depending on N, the maximum value of n we set in equation (2.1), we get a difference of resolution (spectral precision): N corresponding to the space bin on the trigonometric circle and so as the number of different frequencies we consider and the space between each frequency. In this work we will only consider the power of two values for N to get most efficient results.

[5] The Fourier transform is an ideal tool to study stationary signals and processes but music signals as many physics processes are non stationary, evolving through time, it is not an appropriate approach [Hans G. Feichtinger, 1998]

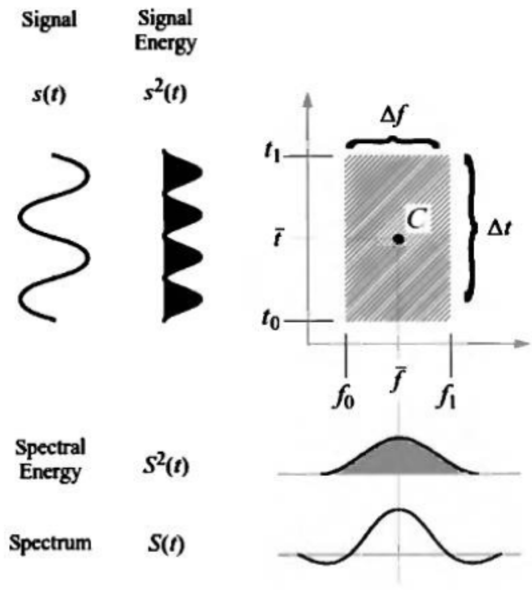

Figure 2.1: Heisenberg box (from [Loy, 2007], p.456))

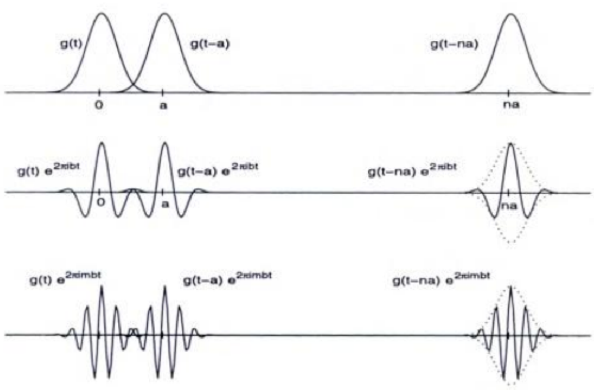

Figure 2.2: Dennis Gabor's elementary functions shifted and modulated copies of a single building shaped block g (from [Hans G. Feichtinger, 1998])

The main result with Dennis Gabor's work is the fact that any signal may now be observed as a set of small objects called grains which are atomic points of bi-dimensional coordinates (time, frequency) of the signal. As we said before, Dennis Gabor wanted to develop a theory of communication based on signals. However, it was less known than Shannon's theory of communication, he was able to develop tools to observe microscopic changes in signal as useful for music generation as for studying quantics physics events.

A grain is a temporal object from 1 ms to 100 ms with a linearly-varying frequency. It can be easily built by multiplying a truncated sinewave magnitude by an envelope (generally gaussian). An example of a grain is on figure 2.3 and a set of grains with silence as a function of time on figure 2.4.

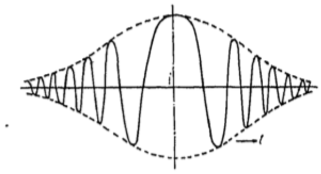

Figure 2.3: Grain with Gaussian shape (from [Gabor, 1946])

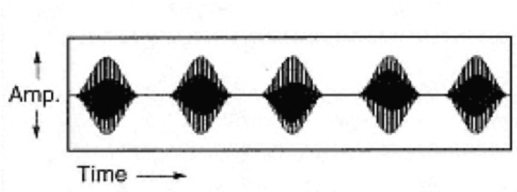

Figure 2.4: A stream of grains with a Hanning envelope (from [Roads, 1995])

In our work, after a few tests we have finally kept the gaussian envelope, however, it it possible to modify many parameters of the grains with the envelope: duration, amplitude and the shape of the envelope (gaussian, attack-decay-sustain-release, excerpt of a song envelope, random, etc...). Figure 2.5 recaps how a granular synthesizer works.

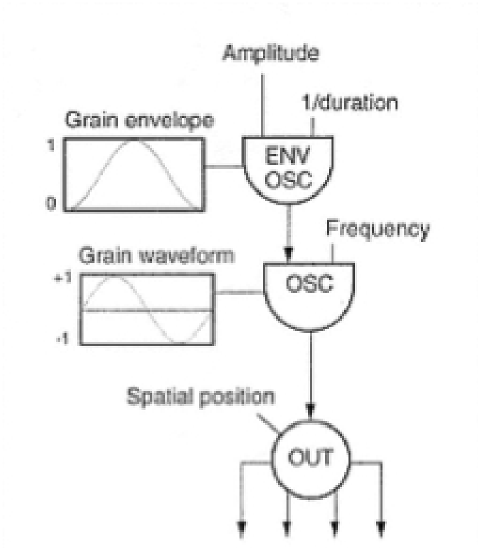

Figure 2.5: A granular synthesis instrument example (from [Roads, 1995])

2.3 Application

In our assessment, we try to get an output in real time. We could have used existing softwares using granular synthesis (typically Max/Msp patchs alike Norbert Schnell's works from Ircam) and try to upgrade them but we believe that we have more options by developing a software from scratch and furthermore a more thorough understanding on the implied processes. We have used a library named portaudio which creates threads for streams. As a thread is an independent process which is supposed to work in real time, we can't expect it to make a lot of calculations. In fact, we have defined the synthesis thread just to store on the output a predefined sine table which is updated by the software whenever it is needed. We have developed a general user interface in C++ with Qt and qwt library to calibrate our synthesizer and check our analysis tools outputs represented in figure 2.6[6].

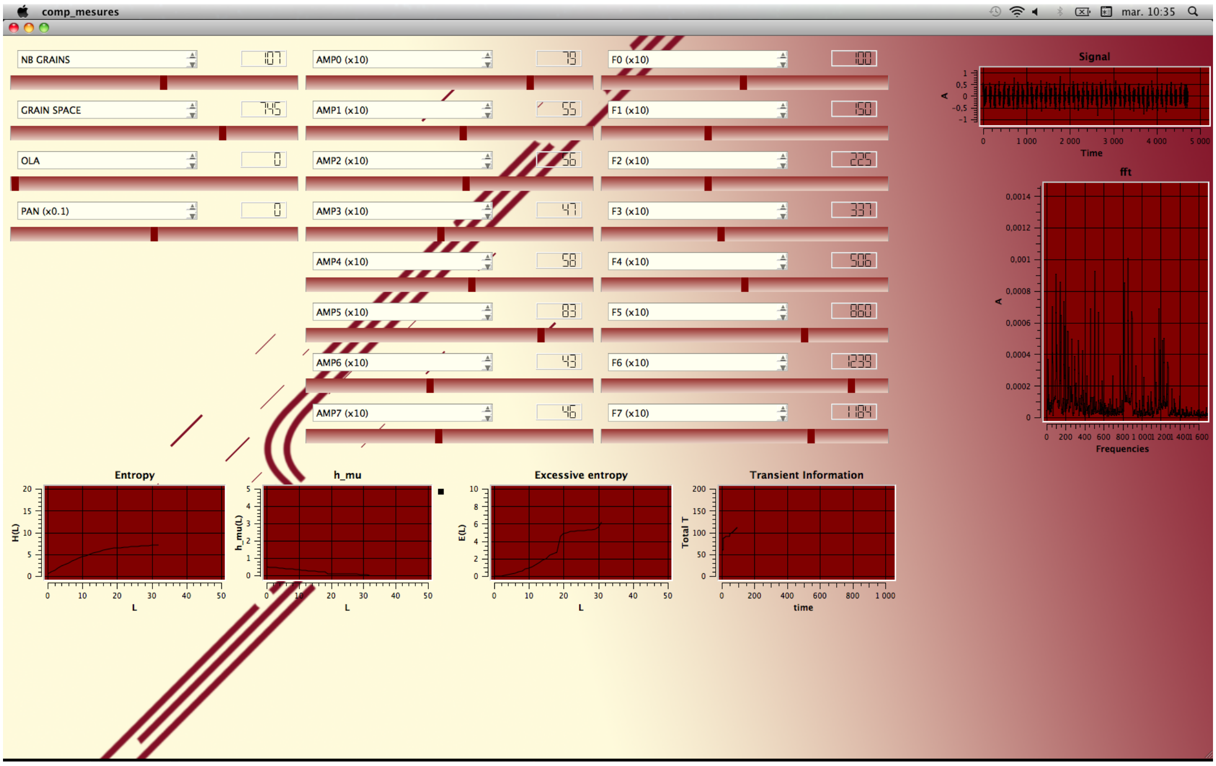

Figure 2.6: Our granular synthesizer general user interface

The output stream stores the values from the finite length table size. The length of the table size is in fact defined by the number of grains per second multiplied by the sample rate and divided by the value of the smallest frequency evaluated. The result is a repetition of the same signal with the same number of grains by seconds each second till new parameters are sent to the synthesizer.

[6]Here is an example after a few manipulations to make the signal more complex by modifying the frequencies and to obtain a more important transient information: at the first step each frequency is the double of the previous one and only one frequency can be modified at a time as each time a button is displaced signal values are recalculated and evaluated by the analysis tools.

Chapter 3

Ideas, methods and results

In the first two chapters, we have developed tools able to analyse and synthesize complex sounds. In this chapter, we will present some ideas on f (the artificial intelligence part of our architecture supposed to match complexity) with results as we had real time measures trouble to concretely develop an f able to match complexity.

In this chapter we have developed two approaches: we have tried to analyse complexity sounds from sequences of hand written numbers or obtained via the synthesizer (Q followed by P) and we have also tried to measure and reduce time analysis on real input data.

3.1 Matching

Artificial intelligence is conceived as if it were an alien arriving on earth and hearing music would decide to try playing with it. It would imply some memory about the musical structure but also a total freeness about harmony rules and sounds panels. Ideally, the program would be an answer to Ada Lovelace's test who argued "until a machine can originate an idea that it wasn't designed to, it can't be considered intelligent in the same way humans are".

Ideally, a "clever" machine would play adjusting complexity to follow as much as possible Wundt's (a psychologist) scheme about equilibrium between reward and punishment to avoid utter boredom from the human musician as in figure 3.1.

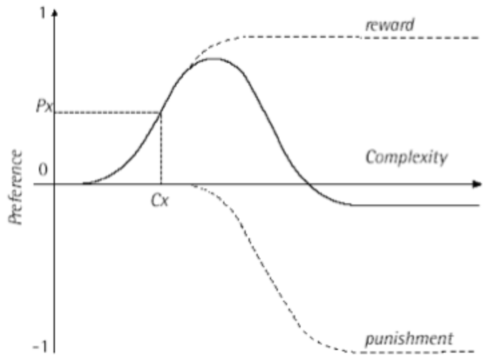

Figure 3.1: The Wundt curve for the relation between music complexity and preference [Streich, 2006].

There are diverse software able to play in interaction with humans, there exist multiple solutions: the machine can wait until it has sufficient information to make decisions, it can play very simple sounds to start a small interaction (but it would imply a previous knowledge on what makes a sound simple) and it can also copy the input signal at the beginning as a mimetic process and try to change its paradigm when it has sufficient data. As we think a musician wouldn't like to play alone too long and would appreciate the surprise of change in the machine, we think that this approach is the most interesting one.

Formally, f as described by Tim Blackwell's modular architecture approach is supposed to get a set of complexity measures from P the analyser and ideally decides the different parameters of Q grains. We first tried to find a simple matching between P and Q. We take each time t, a snapshot of a window of the signal and try to match a complexity knowing the previous signal windows if there are.

The main trouble is the fact that any time we get a new window to analyse, the size of the dictionary $\mathcal{A}^L$ grows up and as does the duration of computable measures. If we want a true real time interaction, we must pay attention that the the output stream thread might not be overcome. As a start, we simplify our initial scheme. We measure the input stream and play as an output stream the same signal to the fact that no solution is found to play without unwanted stream stop. Ideally, the calculus of the complexity should be in a separate thread to avoid all resources to be taken by the measures. As we will see in the measures, the excess entropy may vary quite quickly but always depending on previous knowledge in the dictionary. Another idea is to define different levels of complexity able to catch smaller and larger patterns.

Figure 3.2: Three decorrelated signals: a complete ordered signal, a complex signal and white noise.

Figure 3.3: A subdivision of a white noise signal

Figure 3.2 may be an illustration of chapter 1 about pure tone and white noise complexity assuming pure tone has a completely ordered structure. The purpose of figure 3.3 is to show that different complex structures can be interlinked. As we need to define simpler structures to be computable, we think that we need to keep the links between different levels of complexity. In [Blackwell et al., 2012], Tim Blackwell declares that musicians may develop a macro-structure without being aware of its further development yet we cannot expect from machines a self organisation process from decorrelated signal windows, which has prompted us the definition of complexity levels. As we will observe in the section dedicated to results, we are currently stuck with this way of organisation due to the massive data the machine needs to process.

3.2 Methods

3.2.1 Analysis

In this section we present the different steps we have computed to obtain complexity measures. Concerning the real data complexity measures, we start by storing the input signal in a tab. At each period t, we take a Fourier transform of a bounded window. As we have 44100 samples by second, we have chosen a number of 441 frames per buffer to get 100 chunks per second. From these chunks, we use a median filter to avoid unwanted glitch or errors in the input. By taking only one value about ten, we reduce the number of chunks to ten per second[1]. Then, we apply successively a fast fourier transform (fft) on each chunk and store the different amplitudes values to a new tab which is then given to our analyser. During this step, the energy of the signal is calculated as in formula (3.1).

[1]A median filter has an advantage compared to the mean filter: if an error locally gives some unexpected values, the mean filter will reduce it but the median filter will consider it as extreme values and won't capture it.

Once it is done, we normalise the frequency amplitudes by dividing each of them by the energy.

$$e = \sum_{i=1}^{M} a_i^2 \tag{3.1}$$

where a_i represents the different amplitudes of the fft signal and M the number of different amplitudes in the signal(depending on the size and number of fft).

We must explain that in theory the analyser should take all signal values given to it. However, as measures tend to show that it generally takes around one or two seconds to analyse two chunks by second after two minutes of playing, we wouldn't be able to analyse longer musical period in real time. However relevant or not the results we get, we have with these two chunks 200 ms of spectrum data each second. Depending on the size of the fft, a minimum constant is then defined to put a threshold. Below this threshold, all the data are put to zero. After this step, we are trying to separate as much as possible the different frequencies. We calculate the local maximum around values bounded by zeros to only get the peaks of the signal: if the signal is a pure tone, we wouldn't like it to be considered as a set of local frequencies by the dictionary[2]. The amplitudes stored in the tab representing the spectrum data are then separated in a predefined number of amplitudes bins separated by a logarithmic scale separating a few different high amplitudes but without distinguishing that well lower values. We then obtain our symbols for the dictionary. A summary of the previous steps from signals to symbols is represented in figure 3.4. The number of windows represented is only here to illustrate the model and not its parameters.

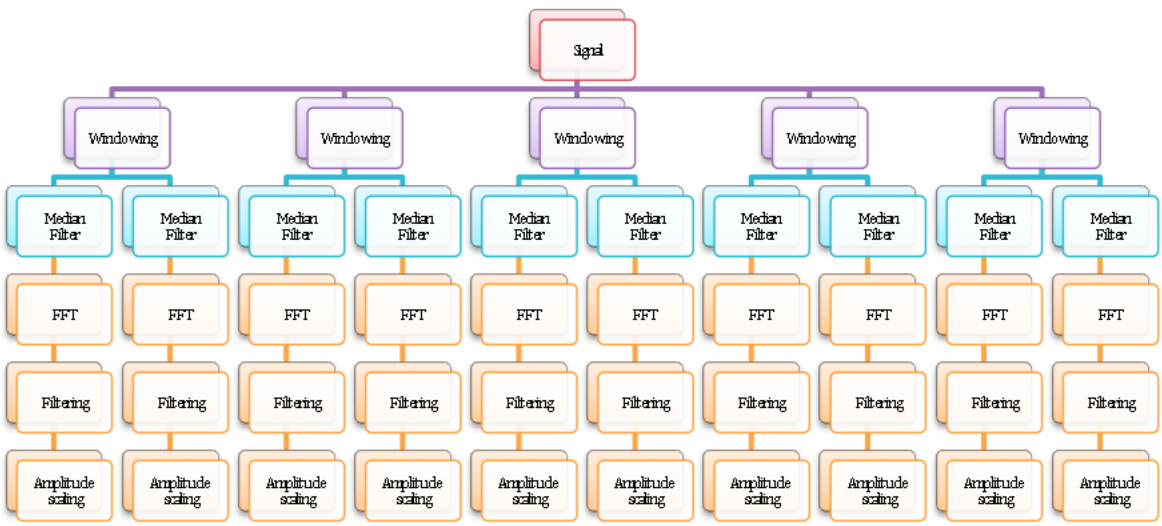

Figure 3.4: Model from signal to symbols to give to the dictionary

Defining a sufficient max order possible of L (200), we only play with the size of the fft and the number of amplitude we consider. We have modified these parameters to see what

[2]In psychoacoustics, it is considered that the human brain does the same by applying an auditive mask cancelling irrelevant data.

is the best way to catch a sufficient number of values to define a Kolmogorov complexity and an excess entropy with a quite low time to measure it. We will show our results in the results section and the ideas we got to go further.

3.2.2 Synthesis

Our goal should ideally be to catch from Q a set of parameters corresponding to a complexity. That is also why in the previous chapter, we have connected our analyser to our synthesizer. The goal should be to get $Q^{-1}(C) = \alpha$ with α the parameters of the grains synthesizer when the complexity button is moved to C by f.

The method we could use, given a set of parameters, could enable us to obtain a set of complexity measures as seen in chapter 2 and to try and match each parameter change to each excess entropy and Kolmogorov change: if the amplitude or frequency x is set to y, getting the new complexity. It implies a record of all complexities in a memory, a learning step. The first criticism we can formulate is the number of data we would need in modifying each parameter one by one, a second would be to consider the fact that with real data, a set of parameters will change at a time. Last but not least, as Chaitin discovered in the 1960's (cf. Introduction), a set of real data can have different measures results as we restart the experiment. As we will see in the next part, an excess entropy measure doesn't mean anything in itself but only compared to other measures to catch the evolution through time. It is then quite difficult to define a deterministic method to observe the change implied by an input complexity to match parameters.

3.3 Results

3.3.1 Analysis

In this subsection, we will speak about the results we got from input real signal as in subsection 3.2.1.

We have started by measuring white noise and pure tone to be sure that our models were corresponding to reality and to catch what the best parameters are to measure data. We have considered a power of two numbers of fft and amplitudes and we shall call them M and N for $|A| = 2^M$, $|\mathcal{F}| = 2^N$. Each time, we have analysed the equivalent of one minute of signal and tested two realisations.

We have considered four linked parameters: excess entropy, transient information, Kolmogorov measure, execution time[3]. The software starts measuring when it detects sounds.

[3]The execution time is computed starting with the signal obtained at the last step in figure 3.4 with log scaled amplitudes, computing Lempel-Ziv compression algorithm to update the dictionary and obtain our measures.

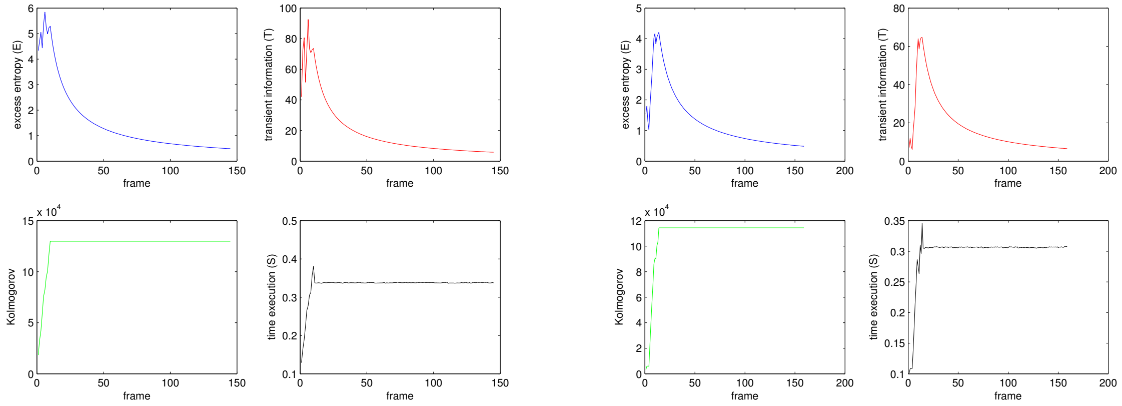

Figure 3.5: White noise measures, N = 6, M = 2, test 1 and 2

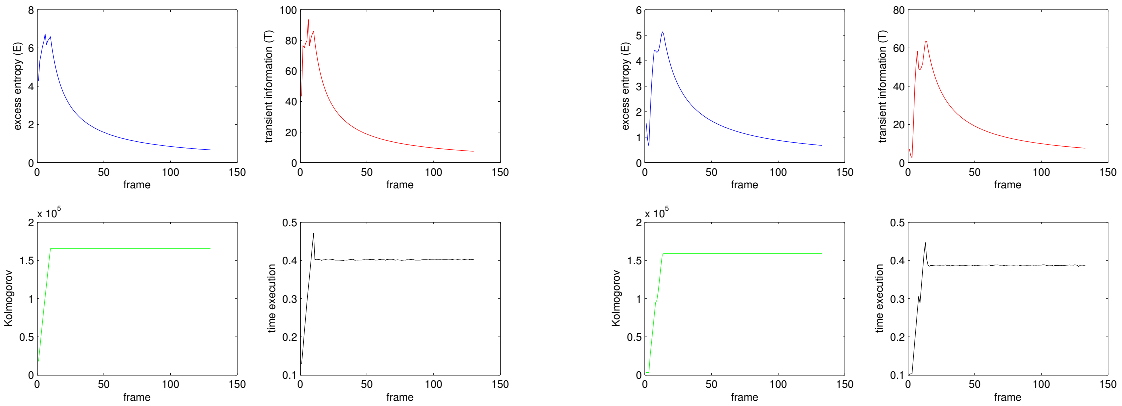

Figure 3.6: White noise measures, N = 6, M = 3, test 1 and 2

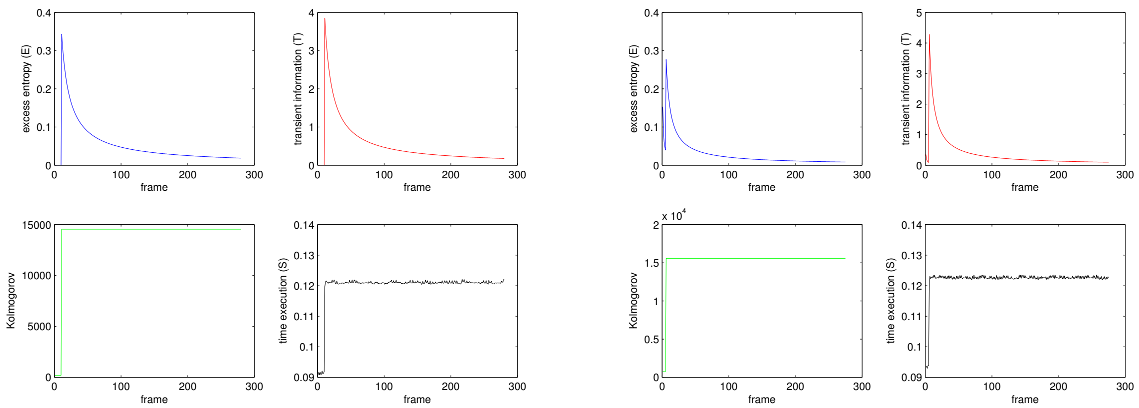

Figure 3.7: White noise measures, N = 6, M = 4, test 1 and 2

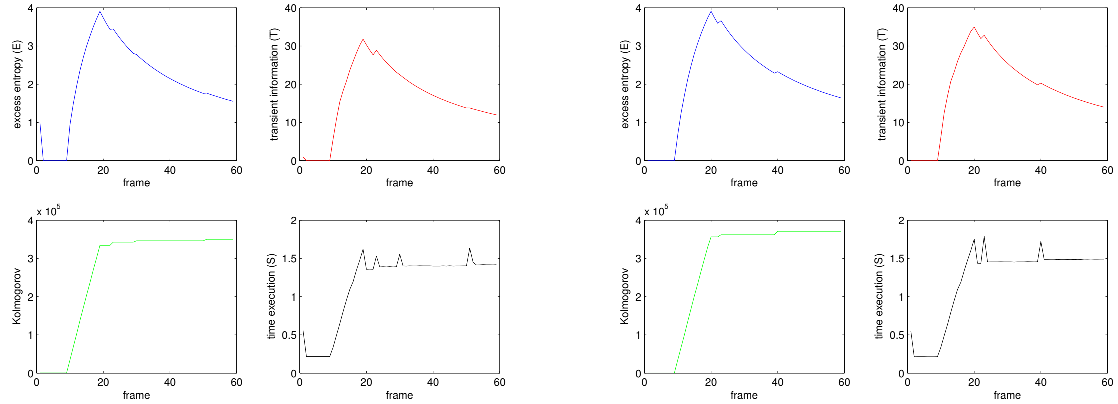

Figure 3.8: White noise measures, N = 8, M = 3, test 1 and 2

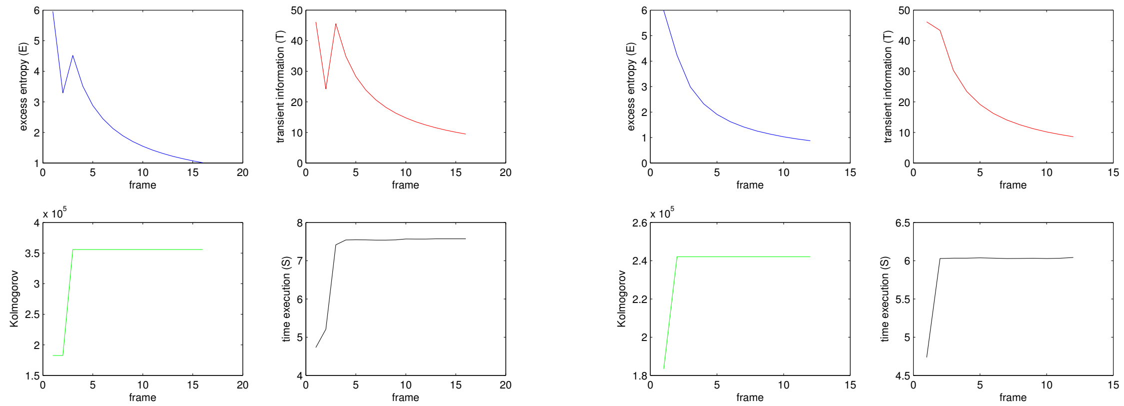

Figure 3.9: White noise measures, N = 10, M = 3, test 1 and 2

Above we can see a lot of data to analyse: as predicted white noise seems to converge to zero with a very quick constant K. Depending on the number of fft and amplitudes considered, we get different shapes but it is quite interesting to see that even if in each figure the two tests have on the whole the same shapes they have different amplitudes in each experiment. It means that excess entropy isn't a result in itself but depends on the experiment conditions. We have also noticed more or less the same shape with different factors in all tests between E and T. We will show further down some results with a comparison between these results to match the difference between these two parameters. We have considered measuring execution time. As you can see in figure 3.9, we get very few frame results because each analysis took approximatively six seconds in comparison to 0.125 seconds in figure 3.7. This is due to the fact that with a larger fft, you get a larger signal given to the dictionary and potentially more different values to treat. Let's see now some results with pure tones.

In chapter 1, we observed that excess entropy was supposed to be the very same between noise and pure tone. However, in figure 3.10, excess entropy and Kolmogorov measure grow through time and results seem to correspond a bit better in figure 3.11 and

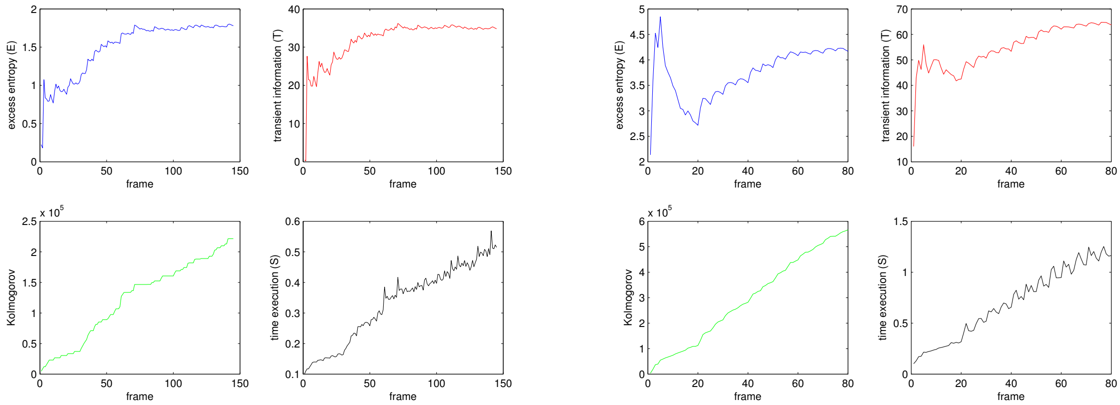

Figure 3.10: Pure tone measures, N = 6, M = 3, test 1 and 2

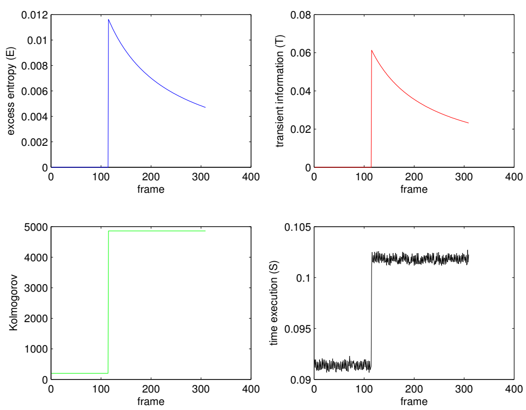

Figure 3.11: Pure tone measures, N = 6, M = 4, test 1

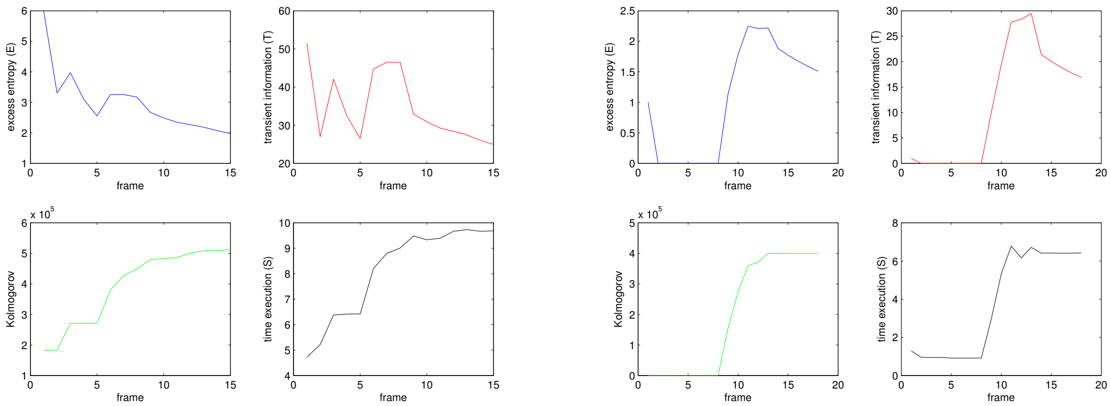

Figure 3.12: Pure tone measures, N = 10, M = 3, test 1 and 2

3.12. The spectral resolution works as a comb, showing all frequencies corresponding to multiples of the smallest frequency detectable. We analyse each time sequences of 200 ms reduced by the median filter to 20 ms. We have then 882 samples: with N=6, we then have 64 bins meaning that the minimum value perceptible is (882/64) 13.7825 Hz. The record has been made with a 440 Hz sinewave and if we check our values, it will fall down into bin 31.92: bin 32 corresponding to 441 Hz and as a consequence the 440 Hz pure frequency shouldn't be captured. We can therefore suppose that as complexity grows up in figure 3.10, we only capture noise. As we explained before, the system takes a lot of process time and overcomes threads. We have decided to be sure that when the system decides to start a new analysis it will hear some sounds. We have recorded from Audacity to a mobile phone all our sounds experiments. The problem might lie on the quality of recording: the frequency band limits of the microphone but also the noise recorded during the experiment. The results are still interesting because contrary to white noise, the noise captured has no regularity and might have some complexity, we can see that for the same parameters of N and M as in figure 3.6, here the analysis time grows up linearly having each time a new sequence of symbols to analyse. As we use a logarithmic scale separating much more higher amplitudes than lower ones, we have in figure 3.11 far fewer errors than in figure 3.10. However, as for other measure tests with the same parameters we only got silence, we can also imply an error during the experiment (noise captured in the room). The last figure shows that in fact a captured pure frequency should on the whole have a decreasing entropy (881/1024 $\simeq$ 0.86 Hz) and fft captures frequencies 439.64 Hz in bin 511 and 440.5 Hz in bin 512, which are very close to 440 Hz pure tone frequency. We can see in figure 3.13 results with a 441 Hz pure tone and with the same parameters as in figure 3.10 corresponding to our expectations.

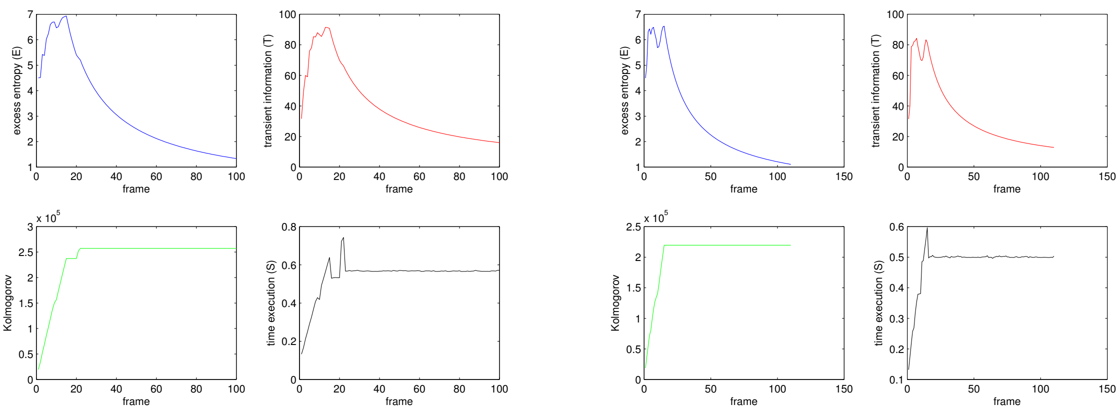

Figure 3.13: 441 Hz pure tone measures, N = 6, M = 3, test 1 and 2

The previous section with pure tones enables us to describe the limits of our system. First, depending on the number of fft bins and amplitudes considered the results might be completely changed. For some very simple sounds, a lower fft could trap data due to its resolution (spectral precision). As we want to play with musicians with more or less complex sounds and a priori without pure tones, our objective here was just to check if our system was corresponding to theoretic measures. If we wanted to get more precision about excess entropy over all sounds in real time, we should have used another feature to be able to get these differences: a tool able to use a large fft size with simple sounds and a smaller one with noise. There exists a feature called spectral flatness which can be used to separate noise from sound. We could initially have applied a large fft and if the sound seemed noisier through time, we would have reduced the fft size or on the contrary if it sounded more musical (a simple sum of a few pure tones) we would have increased the fft size to catch it.

As you might see from figure 3.5 to 3.13, we can only get a growing Kolmogorov complexity as the size of the dictionary may only increase, not decrease. It also implies a dependency in the sound evolution on the previous measures: a sample starting with white noise will have a quicker excess entropy decrease than a sample starting with some complex sounds: complexity measures depend on initial conditions. We cannot use measures as themselves assuming this E corresponds to a specific sound but only considering ΔE and ΔT parameters. As seen previously, there is quite a similitude between E and T and we have tried to test if we could expect some results comparing these two parameters.

We are presenting results here with a whistle consisting in a frequency modulation from 440 Hz to 1320 Hz to observe the limits of our system and then an analysis of our last results in a few songs.

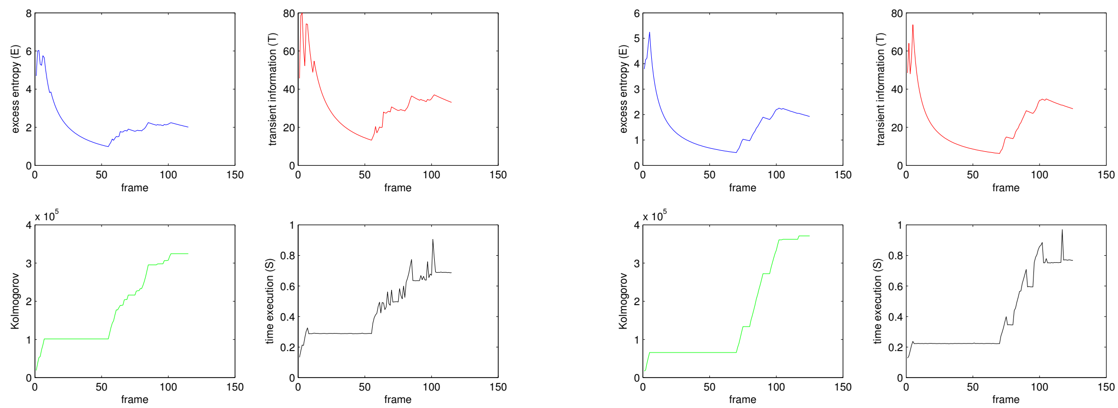

Figure 3.14: A whistle from 440 Hz to 1320 Hz, N = 6, M = 3, test 1 and 2

As you can observe in figure 3.14 and as we have already observed before, our system takes some time to be stable with higher frequencies due to dft. However, we can observe with Kolmogorov that even in this case Kolmogorov measure is stationary through time.

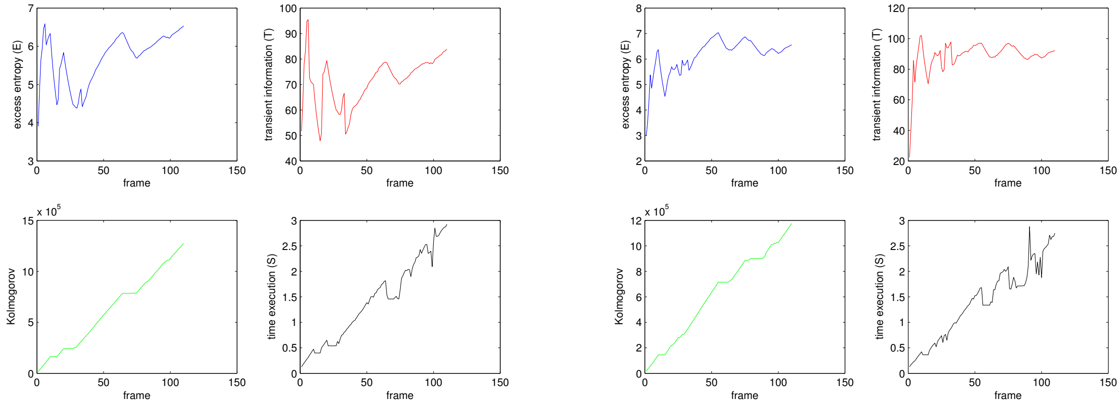

Figure 3.15: Brian Ferneyhough string quartet number 5, beginning, 2'30", N = 6, M = 3, test 1 and 2

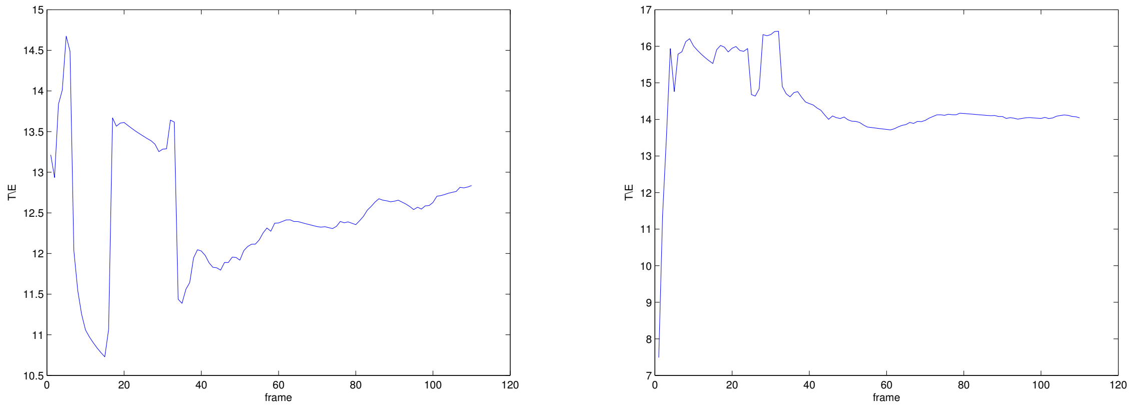

Figure 3.16: Brian Ferneyhough string quartet number 5, beginning, 2'30", transient information divided by entropy, test 1 and 2

Figure 3.16 shows $\frac{\Delta T}{\Delta E}$ Bryan Ferneyhough excerpt. Bryan Ferneyhough works are known for their high complexity level to be mostly unplayable by human musicians. In figure 3.15 and 3.16, we can observe two realisations of measures with the same data. Even if the results in figure 3.15 seem to be quite different, it is interesting to observe that in figure 3.16, the results seem to have some similarities: we have the same $\frac{\Delta T}{\Delta E}$ important evolution between samples 20 and 40. We believe that the complexity of a work may be observed by the variation of these two parameters through time: as transient information measures the difficulty to synchronise and as excess entropy measures the redundancy of the signal, it seems consistent to say that complexity might be apprehended as the variation of difficulty to synchronise to a work while recognising some of its patterns.

In the fields of social psychology, definitions about behaviour percepts are used to look for a musical attitude of consensus (how far the reaction of a musician is comparable to that of others), of consistency (how far the musician reacts in the same way given the same individual musical stimulus) and of distinctiveness (how far the musician reacts in the same way to differing stimuli) [Blackwell and Young, 2014] [Kelley, 1972]. We believe that T and $\frac{\Delta T}{\Delta E}$ might let us observe and apply these parameters as even if E results may differ through experiences, $\frac{\Delta T}{\Delta E}$ is mostly the same showing some difference with the same important pattern detections.

We can check results with some other works, we have chosen popular songs: a Django Reinhardt's song known for its harmonic complexity and a Serge Gainsbourg's reggae song for its quite repetitive tune.

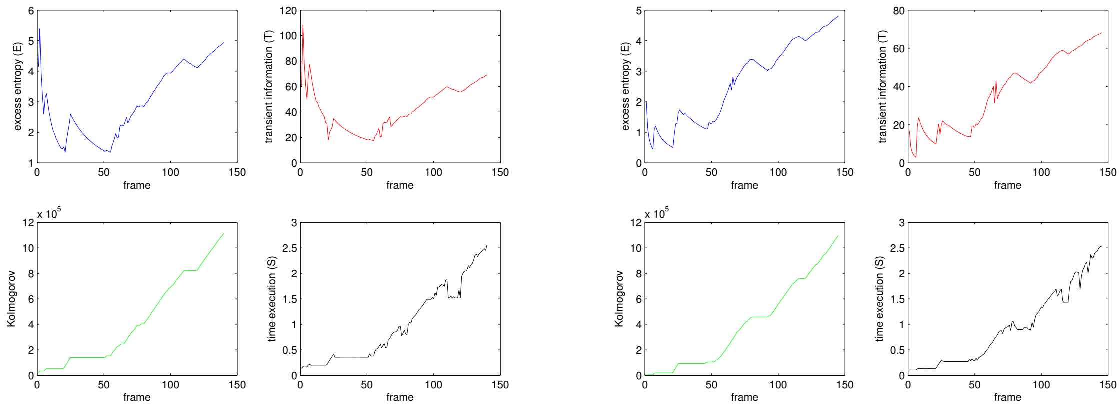

Figure 3.17: Django Reinhardt, Les Yeux Noirs (1940), 2'18", N = 6, M = 3, test 1 and 2

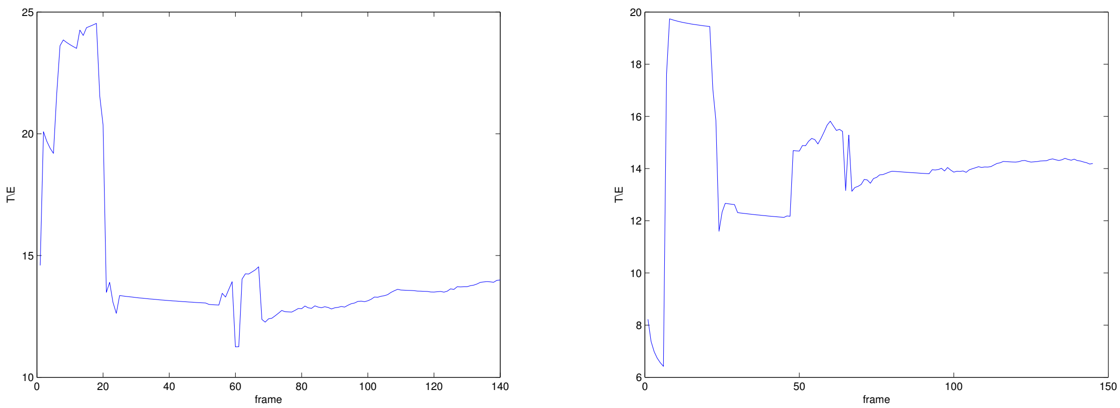

Figure 3.18: Django Reinhardt, Les Yeux Noirs (1940), 2'18", T divided by E, test 1 and 2

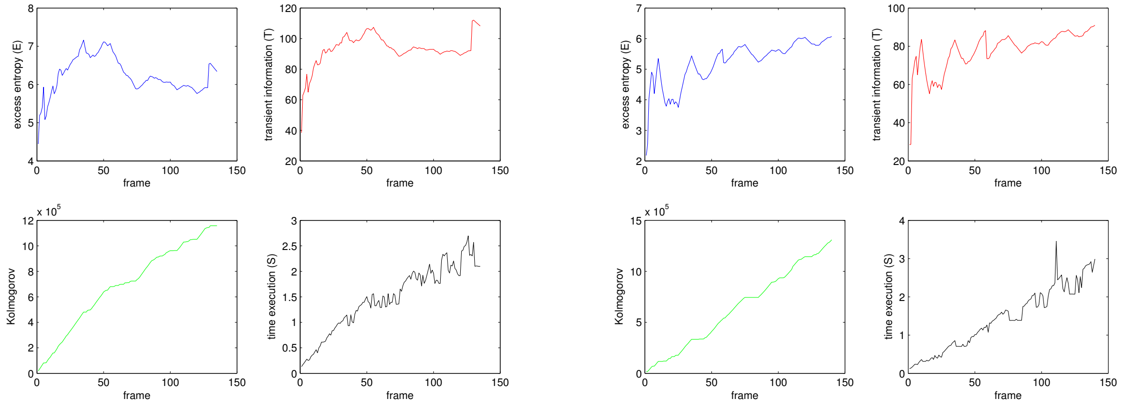

Figure 3.19: Serge Gainsbourg, Aux armes et cætera (1979), 3'10", N = 6, M = 3, test 1 and 2

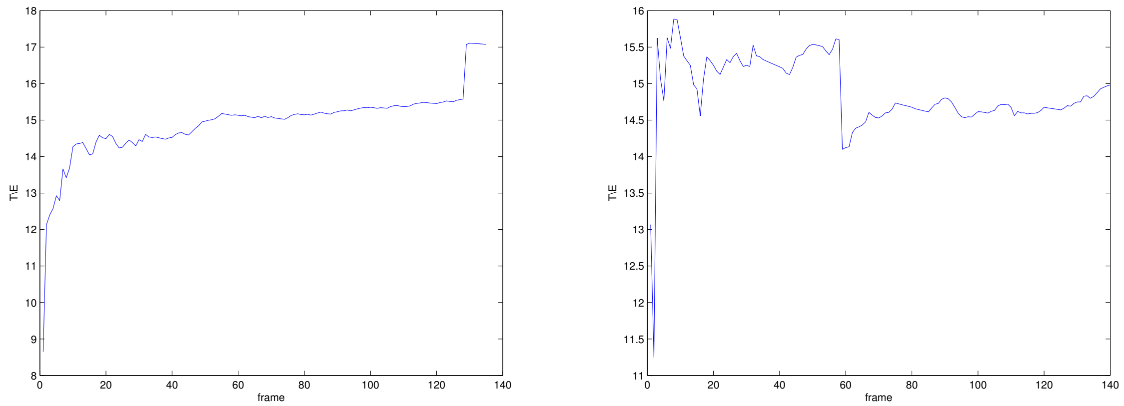

Figure 3.20: Serge Gainsbourg, Aux armes et cætera (1979), 3'10", T divided by E, test 1 and 2

We can observe in figure 3.18 that complexity seems to vary importantly from time to time when solos takes place whereas complexity is quite always the same in figure 3.20 as expected.

The way the measures are taken does not allow us to declare that Reinhardt's song or Ferneyhough's work are more complex than Gainsbourg's but that these works contain in themselves more complexity variations than the latter.

3.3.2 Synthesis

In this subsection, we will explore some results we got from synthesis measures. While varying parameters, we have tried to find which ones are the most efficient to reach expected excess entropy variations.

We have tested quite simple parameters: we have defined eight frequencies as $f_0 = C$, $C \in \mathbb{N}$ and $f_n = D.f_{n-1}$ or $f_n = D^n.f_0$, $1 < D \leq 2$.

As ancient Greeks used to consider that the most harmonious parameters were those which happened in a double frequency ratio, we have started with $D = 2$ and $C = 100$ Hz. We then have taken Pythagorean frequency intervals fit to put in an octave to compare differences with $D = 1.5$.

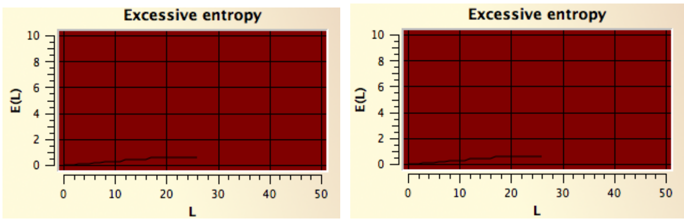

Figure 3.21: Granular synthesiser excess entropy with C = 100 Hz, D = 2 (left) , D = 1.5 (right)

We obtain quite similar results. We cannot observe differences in figure 3.21 concerning these two harmonious sounds. As we are testing with quite low frequencies, we are trying to see if a higher frequency spacing will improve these differences with C = 1000 Hz.

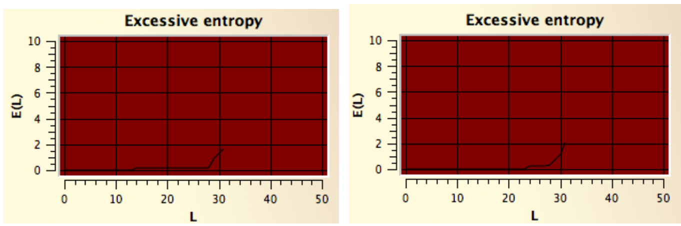

Figure 3.22: Granular synthesiser excess entropy with C = 1000 Hz, D = 2 (left) , D = 1.5 (right)

In figure 3.22, we get some distinctions between the two excess entropy measures. However, as the maximum order of L isn't assigned by hand but calculated with a stop criterion about entropy maximum, we cannot make out much difference between harmonious and inharmonious sound. As we consider the fft, shifting the values to high frequencies allows us to distinguish more values of the signal as a complex sound more than as a noise. We guess that shifting a frequency will display new geometries of symbol sequences, which will enables us to obtain a more important excess entropy whereas just modifying frequency amplitudes will entail few changes as in figure 3.23 due to a logarithmic scale and energy repartition.

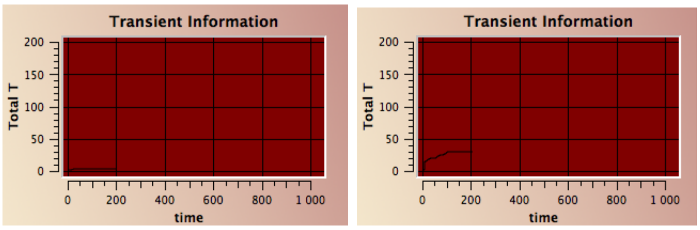

Figure 3.23: Granular synthesiser transient information after 20 random amplitudes (left) and after 20 frequencies (right)

In figure 3.23, we can also observe a stabilisation of T through time. We have applied the idea of frequency shifting and tried to see if we could easily obtain transient information modifications. It seems that increasing space between frequencies and decreasing it afterwards let us obtain these results: i.e. transforming almost a noise into a sound and then transforming it again into almost a noise.

As we work with a granular synthesiser we can also modify the number of grains per second and the space between grains. The number of grains per second has no implication in itself to excess entropy modification as it has no influence on fft signal. However, the more grains per second we have the more space between grains we can add without completely cancelling the signal. In chapter 2, we explained that the table size was defined by the number of grains per second, the sample rate and the lowest frequency detectable. The number of grain spaces separate the table size in a grain spaces number and cancel a space size number of values, space size defined as a constant S, with $S = \lfloor 10\frac{\text{grains space}}{\text{grains per second}} \rfloor$.

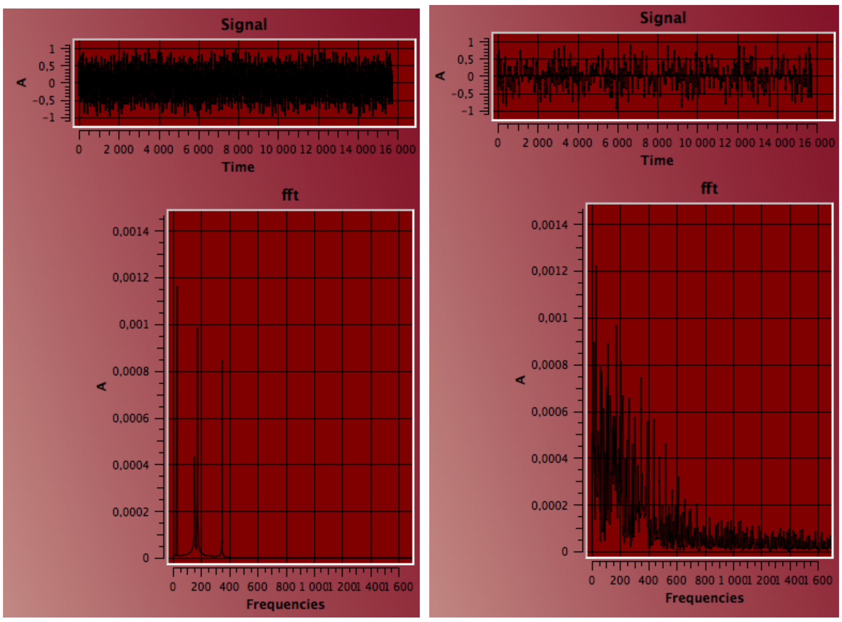

Figure 3.24: Same signal with four frequencies and 100 grains without grains space (left) and most values cancelled due to grains space (900) (right)

In figure 3.24, we can observe large signal and fft difference due to zeros in the signal, which implies much more pseudo sine values. As a result, excess entropy and transient information vary a lot through these parameters. Yet, after testing a growing and decreasing grain spacing with low frequency space and large frequency space, we get no consistent results.

In the next test we choose to have $f_0 = C$, with C = 100 Hz and then C = 1000 Hz associated with this new formula $f_n = (n+1).f_0$.

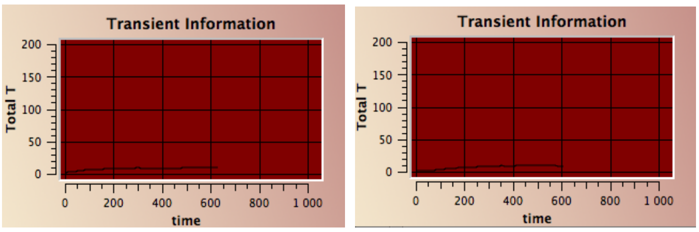

Figure 3.25: Low frequency spacing (100 Hz) with 200 grains and starting with a decreasing then growing number of grain spaces on the left and the contrary on the right

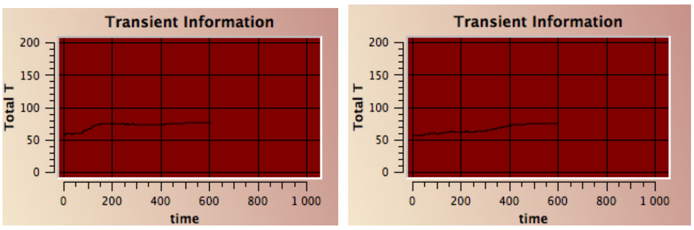

Figure 3.26: High frequency spacing (1000 Hz) with 200 grains and starting with a decreasing then growing number of grain spaces on the left and the contrary on the right

We observe a few differences concerning transient information variations between high frequency spacing and low frequency spacing in figure 3.25 and 3.26 and, as expected, with a larger frequency space we obtain a better differentiation. However, we cannot deduce general laws which would enable us to get precise transient information or excess entropy, whenever needed, since the results we obtained seem to evolve more slowly as the number of data in the dictionary grows up.

As a conclusion, we haven't got any tools yet to determine how to move from an excess entropy or a transient information to the next one as freely as we would like to. We have thought out three solutions: as we get all signal windows and never remove them, we could associate each fft to a complexity as we measure excess entropy and transient information; we could also measure fft variation and associate each result to complexity variation: whenever getting a complexity variation, we could observe the variation between the latest fft measures for each bin and determine the next fft to play applying the same variation to the last fft known.

The last solution we have thought about is lighter as we do not need to remember any fft signals: we could just take symbols addresses in the dictionary, each one of them being associated to a probability and building an N size chunk from scratch by adding, from highest L order possible to lowest possible, a set of chunks corresponding to the product of probabilities as close as possible as expected: since the larger size chunk would correspond to low frequencies where human ears cannot distinguish much frequency differentiations, we could have a tool able to play thus a large spectrum of different signals; but this would require a decision about the probability of each set of symbols, expected from a set of complexity measures.

Since this last option would make us interpolate fft values from symbols, we could use a learning step with a fixed number of amplitude frequencies parameters associated to each number of grains per second and all grain spacing possible to match these last two parameters to a sequence of distinguishable symbols. The created chunks could then modify, as needed, the number of grains per second, the grain spacing, the frequencies and amplitudes required. These are only prospective ideas.

Conclusion

In this work, we have dealt with the notion of complexity. The fields of information theory has been made clearer to us and a few results we proved consistent have been found. As we have shown, the methods we developed are far from being perfect (low fft size, low number of frequency amplitude differentiation, etc.) but these problems are linked to common trouble in complex systems: real-time management, thread parallelism (input/output streams) and massive data processing.

It has also taken us a long time at the beginning of our assessment to develop a general user interface additive granular synthesizer from scratch. We had tested a lot of parameters concerning envelopes (ADSR models) which are not presented here because they were judged secondary to the main results implied by complexity measures.

Due to the number of the different tricks encountered, a playable P-f-Q live algorithms architecture could not be completed but we believe that our results remain in themselves encouraging for the complexity music future. Based on these results, our next goal should be to focus on the artificial intelligence part to be able to interact and check machine reactions to real musical propositions, repetitions or novelties.

Even though complexity measures seem "complex" and take a long time to compute, their purpose is to find out the inner rules in music without cultural a priori and supposing these rules are quite simple, based on a novelty-redundancy ratio to enhance new musical propositions. Although, our results have been obtained through a quite old computer, we believe that we would need a powerful server to make this software run in real-time and this should represent the future computer music quest which would allow us to overtake Ada Lovelace's test with independent creative machines.

Appendix A

Appendix

A.1 LZ78 algorithm and Kolmogorov estimation

```
/* count LZ78 segments */
Clear dictionary;
w = λ;
Kest = 0;

while (more input)
      C = next symbol;
      if (wC in dictionary)
         w = wC;
      else
         Kest = Kest+1;
         add wC to dictionary
         w = λ;
      endif
endwhile

if (w!=λ)
      Kest = Kest+1;
endif

return Kest;
```

Figure A.1: Lempel Ziv 78 algorithm and Kolmogorov upper boundary estimation (from [Li and Sleep, 2004])

A.2 Synthesis software structure

```
.
|----- Makefile
|----- README
|----- TODO
|----- bin
|    |----- comp_mesures
|----- include
|    |----- PaGrains.h
|    |----- PaGrainsData.h
|    |----- PaParameters.h
|    |----- Window.h
|    |----- Window.moc
|----- pictures
|    |----- background.png
|    |----- background2\ copie.png
|    |----- background2.png
|----- src
     |----- PaGrains.cpp
     |----- PaGrainsData.cpp
     |----- PaParameters.cpp
     |----- Window.cpp
     |----- main.cpp
```

Figure A.2: Structure of synthesis software

```
#ifndef PAGRAINS_H
#define PAGRAINS_H

#include <cstdio>
#include <cstdlib>
#include <cmath>
#include <ctime>
#include <fftw3.h>
#include "portaudio.h"

class PaGrains{

    public:

    PaGrains();
    ~PaGrains();

    double * get_amplitudes();
    void set_amplitudes(double * values);

    double * get_sine();
    double get_sine(long int index);
    void set_sine(double * values);

    long int get_table_size();
    void set_table_size(long int value);

    unsigned short get_grains_per_seconds();
    void set_grains_per_seconds(unsigned short value);


    private:

    double * amplitudes;
    long int table_size;
    unsigned short grains_per_seconds;

};

#endif /* PAGRAINS_H */
```

Figure A.3: Grain definition

```
#ifndef PAGRAINSDATA_H
#define PAGRAINSDATA_H

#include "PaGrains.h"

class PaGrainsData : PaGrains
{
    public:

    PaGrainsData();
    ~PaGrainsData();

    PaGrains * old_grains;
    PaGrains * new_grains;

    unsigned short get_grains_space();
    void set_grains_space(unsigned short value);

    double get_pan();
    void set_pan(double value);

    private:
    unsigned short grains_space;
    double pan;
};

#endif /* PAGRAINSDATA_H */
```

Figure A.4: Grains evolution management

Bibliography

[Blackwell and Young, 2014] Blackwell, T. and Young, M. (2014). *Live Algorithms for Music : Can Computers Be Improvisers ?*, volume 1 and 2 of *Oxford Handbook of Critical Improvisation Studies*. Oxford University Press, New York.

[Blackwell et al., 2012] Blackwell, T., Young, M., and Bown, O. (2012). *Live Algorithms*. Springer, Berlin.

[Crutchfield, 2003] Crutchfield, J. P. (2003). *Art and Complexity*, chapter 3. Elsevier.

[Crutchfield and Feldman, 2003] Crutchfield, J. P. and Feldman, D. P. (2003). Regularities unseen, randomness observed: Levels of entropy convergence. *CHAOS*, 13(1):25–54.

[Gabor, 1946] Gabor, D. (1946). Theory of communication. *The Journal of the Institution Of Electrical Engineers*, 93(3):429–457.

[Hans G. Feichtinger, 1998] Hans G. Feichtinger, T. S., editor (1998). *Gabor Analysis and Algorithms, Theory and Applications*. Birkhäuser Boston.

[Kelley, 1972] Kelley, H. (1972). *"Attribution in Social Interaction" in Attribution: Perceiving the Causes of Behaviour*. General Learning Press, Morristown NJ.

[Li and Sleep, 2004] Li, M. and Sleep, M. R. (2004). Melody classification using a similarity metric based on kolmogorov complexity. Proceedings of the Sound and Music Computing Conference (SMC'04).

[Loy, 2006] Loy, G. (2006). *Musimathics — The Mathematical Foundations of Music, Vol. 1*. Cambridge: MIT Press.

[Loy, 2007] Loy, G. (2007). *Musimathics — The Mathematical Foundations of Music, Vol. 2*. Cambridge: MIT Press.

[Mitchell, 2011] Mitchell, M. (2011). *Complexity: A Guided Tour*. Oxford University Press.

[Morin, 1977] Morin, E. (1977). *La Méthode 1. La Nature de la Nature*. Seuil.

[Roads, 1995] Roads, C. (1995). *The Computer Music Tutorial*. MIT press.

[Streich, 2006] Streich, S. (2006). *Music complexity: a multi-faceted description of audio content*. PhD thesis, Universitat Pompeu Fabra.

[Xenakis, 1955] Xenakis, I. (1955). La crise de la musique sérielle. *Gravesaner Blätter*, 1:2–4.

[Xenakis, 1963] Xenakis, I. (1963). *Musiques Formelles.* La Revue Musicale.