QUANTIFICATION OF DETERMINISM IN MUSIC USING ITERATED FUNCTION SYSTEMS

BRIAN MELOON

JULIEN C. SPROTT

University of Wisconsin at Madison

ABSTRACT

This paper proposes a novel technique for exhibiting and quantifying the determinism in music. A written score of music is modeled as a dynamical system employing an iterated function system to generate a picture from the music. This picture is then analyzed using methods of chaos theory and time-series analysis to quantify the determinism. Comparisons with random and chaotic control data and with some algorithmic compositions are made. The method might be useful for cataloging different musical styles or perhaps even testing authenticity of musical compositions.

INTRODUCTION

This paper investigates the use of a novel technique to search for determinism in written music and to quantify its extent. We assume that such determinism will exist since composers use rules, both consciously and subconsciously, when composing. Our approach works from the original score, instead of with an aural representation, as previous such approaches have taken. To illustrate the method, we have limited our scope to a subset of Baroque and Classical compositions, primarily because of the easy availability of appropriate, machine-readable, musical data.

METHOD

Step 1: Initial Simplifications

To model a piece of music such as the one shown in Fig. 1 as a dynamical system, we need to make three simplifications.

>>> INSERT FIGURE 1 ABOUT HERE <<<

First, we need to separate the voices. Since our method requires a single temporal sequence of numerical values, we split the music into individual melody lines, called voices. This limits the pieces we can simply analyze to those which can be logically split into such voices.

Second, we ignore all octave information; we reduce the notes to their pitch class, a letter from A to G-sharp, representing the twelve notes in an octave. We have made no attempt to distinguish between enharmonic pitch classes. This is done to make our approach independent of the pitch range for a given piece. By reducing to pitch classes, we can compare different pieces using the same approach. Furthermore, this method compresses the data range, thus providing reliable results with less data.

Third, we ignore duration information. This is done to simplify the analysis. The method could be easily extended to work with durations by simultaneously analyzing pitch and duration in a higher dimensional space. This approach would undoubtedly give more interesting results, since rhythmic structure is very important in music.

Applying these simplifications to the piece in Fig. 1 provides the following data streams:

voice 1: A G A B-flat G G A B-flat

voice 2: A A A A G G G G

voice 3: F F F F E E E E

voice 4: D

voice 5: D D

This is the form of the score we shall use for illustrative purposes. However, we must do a further step before the analysis, since the data is still in discrete form.

Step 2: The Iterated Function System

We shall use an iterated function system (IFS) [1] to produce a picture from data in the form above. Our idea is an extension of one used by Jeffrey [2] to generate a picture from the sequence of base pairs in the DNA molecule.

Since the IFS is the most conceptually difficult step, we shall illustrate with a simple example. A more thorough treatment can be found in Jeffrey [3] or Barnsley [1]. For discrete data, which can take on only the values of 1,2,3, or 4 (corresponding, say, to the four bases, ACGT, of the DNA molecule), we would define an iterated function system by choosing a "world" (a simple example is the unit square with each of the four values corresponding to a different corner of the square) and an arbitrary starting point in that world, and making the following rules:

If the next value is a 1, move halfway toward (0,0);
If the next value is a 2, move halfway toward (1,0);
If the next value is a 3, move halfway toward (1,1);
If the next value is a 4, move halfway toward (0,1).

We start at the initial point, and then read in the data, plotting a succession of points. The result is a scattering of points in the plane. The plot represents a trajectory, since the position of each point is determined by all the previous points. The crosses in Fig. 2(a) are meant to show what the successive iterations of this process do to an initial pattern in the plane defined by the large cross that bisects the square horizontally and vertically, with the data sequence 1, 3, 4, 2. Notice that with each iteration the cross moves and becomes smaller. What we are doing is combining the discrete data points to spread the data more uniformly over the plane.

>>> INSERT FIGURE 2 ABOUT HERE <<<

In generalizing this idea to data with more than four discrete values, as is necessary for musical scores, we need more complicated rules. For each data value, we need to write explicit mathematical formulas that map the whole world into non-overlapping subsections of the world. For the simple example above, the rules are:

If input value is 1: $\{x_{\text{new}} = x_{\text{old}} / 2,\ y_{\text{new}} = y_{\text{old}} / 2\}$

If input value is 2: $\{x_{\text{new}} = x_{\text{old}} / 2 + 1 / 2,\ y_{\text{new}} = y_{\text{old}} / 2\}$

If input value is 3: $\{x_{\text{new}} = x_{\text{old}} / 2 + 1 / 2,\ y_{\text{new}} = y_{\text{old}} / 2 + 1 / 2\}$

If input value is 4: $\{x_{\text{new}} = x_{\text{old}} / 2,\ y_{\text{new}} = y_{\text{old}} / 2 + 1 / 2\}$

If we use the subscript *i* to denote an arbitrary one of the data values, we can denote all four rules using the following compact mathematical notation:

$$H\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x/2 \\ y/2 \end{bmatrix} + \Gamma_i$$

where Γ_i is the i^{th} column of Γ, where Γ is a matrix given by

$$\Gamma = \begin{bmatrix} 0 & \frac{1}{2} & \frac{1}{2} & 0 \\ 0 & 0 & \frac{1}{2} & \frac{1}{2} \end{bmatrix}.$$

We can generalize this notion to our 12-valued discrete musical data. We begin with a rectangle of width four, and height three, and partition it into 12 squares of unit length, in a four-by-three arrangement (figure 2(b)). This partitioning was chosen so that the pieces of the partition are as similar in shape to the original rectangle as possible, to avoid building inherent structure into the plots our process will create. This partition also has the desirable property that the pieces intersect only at their boundaries. Next, we mathematically formulate the transformations used for this system to map the whole rectangle into the smaller pieces. Finally, we associate each square with one of the possible musical notes, making no distinction between enharmonic pitch classes. How we do this is arbitrary, since different associations will yield similar results.

For our IFS, if we assign $i = 1$ for the note A, $i = 2$ for the note A-sharp, and so forth, the transformations are:

$$H\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x/4 \\ y/3 \end{bmatrix} + \Gamma_i$$

with

$$\Gamma = \begin{bmatrix} 0 & 1 & 2 & 3 & 3 & 2 & 1 & 0 & 0 & 1 & 2 & 3 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 2 & 2 & 2 & 2 \end{bmatrix}.$$

The IFS process is creating a trajectory through the plane for each voice. Since the interiors of our tiling do not overlap, given an initial starting position on the boundary of the rectangle (we used the point (0, 0), which is the lower left corner), each point to which the IFS can map is sufficient, in principle, to characterize the voice completely. That is, each point represents all the notes that have been played and preserves the order in which they were played. In a sense, we are reverse-ordering the notes, since if we wish to recover the notes from a given point, we start by finding out which note was last played, and working backwards through the sequence of points.

Since computers only have finite accuracy, we cannot expect each point on the plot to reflect the entire previous history of the piece. Each point only gives a short-term history of the voice because the influence of previous notes diminishes as time goes on. This history should be about 35 notes, since we are contracting the rectangles by factors of four and three in the *x* and *y* directions, respectively. It is for this reason that an arbitrary point on the border can be chosen as the starting point. Except for the first few points, the pattern is essentially the same for any starting point.

Using the method described above, the IFS can be used to make plots for each voice which are then superimposed into a single plot. Two examples of such plots are shown in Fig. 3. Figure 3(b) is a piece of real music, Mozart's "Sonata in C." There are approximately 2000 points on this plot. For comparison, Fig. 3(a) shows 2000 randomly chosen notes. As one can see, Fig. 3(a) is much more uniformly filled than Fig. 3(b). Any departure from a uniform distribution of points is evidence for determinism.

>>> INSERT FIGURE 3 ABOUT HERE <<<

The first thing to notice about the output of the IFS process is that we can immediately gain some useful information from it. Each time a given note is played, a dot is placed somewhere in the square corresponding to that note. Therefore, if we know the original IFS scheme, we get an indication of the number of times each pitch class was played, and perhaps identify the main scale that was used in the piece. This is the same information that is contained in a histogram of the pitch classes. The IFS plot in Fig. 3(b) clearly shows a higher density of points in the squares corresponding to A, B, C, D, E, F, and G, which make up a C major scale. The plot is a two-dimensional visual representation of the piece, and it is unique to that piece.

Step 3: Correlation Dimension

To quantify the determinism, we need to measure some property of the IFS pattern such as its dimension. If the points were uniformly spread across the plane and if there were an infinite number of them, they would constitute a two-dimensional object (a surface). More generally, the plots are expected to be fractals with self-similar structure and fractional dimension. There are many different ways to define and calculate the fractional dimension of an object, but one that is commonly used because of its ease of implementation is the correlation dimension proposed by Grassberger and Procaccia [4].

For a data set of size N, with points X_1, ... , X_N, we define the correlation sum as a function of scale size r by:

$$C(r) = \frac{2}{N(N-1)} \sum_{i=1}^{N} \sum_{j=i+1}^{N} \Theta(r - \| X_i - X_j \|)$$

where Θ is the Heaviside step function defined such that $\Theta(x)$ is zero for $x < 0$, and one for $x \geq 0$, and each X_i is a point in the plane representing the i^{th} data point in the IFS pattern. The second summation is taken over only the data points that follow each point in the first sum ($j > i$) to avoid double-counting. The notation $\|X_i - X_j\|$ means the distance between points X_i and X_j. In other words, the correlation sum is the number of pointwise distances less than r divided by the total number of distances, or, equivalently, the probability that two randomly chosen points will be within a distance r of one another.

This discussion suggests what different scales (values of r) in the correlation sum mean. On the large scale, we are measuring the overall distribution of the notes in pitch classes. At smaller scales, we are looking at longer-term information, the relative frequency of motifs recurring. On the order of $r = 10^{-4}$, we are looking at motifs of approximately eight or nine notes.

The correlation dimension is given by the slope of the correlation sum, $\mathrm{d}C(r) / \mathrm{d}r$. For a precisely self-similar fractal this slope is independent of the scale size parameter, r. However, for real data, the slope is seldom independent of r, and a single dimension thus cannot be defined. For this reason, we plot the dimension versus scale size. If the plot has a plateau, the object is a self-similar fractal over that scale range and has a well-defined fractional dimension.

RESULTS

Real Music

A dimension plot obtained as described above for Mozart's "Sonata in C" as shown in Fig. 4 is typical of the curves generated by real music. The curves for Pachelbel's "Canon", Beethoven's "Moonlight Sonata", and Bach's "Fugue in C Major", "Fugue in B Major", and "Well Tempered Clavier, Book1, Prelude 1" are similar. Thus these composers employ a similar degree of determinism in their compositions. Note that the dimension plot has no obvious plateau, and thus there is no unique numerical value that can be associated with these composers.

>>> INSERT FIGURE 4 ABOUT HERE <<<

Artificial Music

While the dimension provides a quantitative measure of the determinism in a musical piece, we still need control data to establish its statistical significance. On way to do this is to create artificial music, using the same notes as the original piece, but shuffling the order in a random or deterministic way. This allows us to remove the determinism originally present in the music or to replace it with a different kind of determinism. In this way, we can compare different types and degrees of determinism. We need to preserve the number of times each note is played, since the distribution of notes will generally affect the dimension curve.

We chose three main methods for creating artificial music. First, we created a non-deterministic, unpredictable piece by randomly shuffling the notes in each voice. Second, we created a deterministic and highly predictable piece by making a histogram of the notes in each voice, and removing them one by one, taking an A if there was one, then taking an A-sharp if there was one, then taking a B if there was one, and so forth. This method produced a strongly deterministic composition akin to simply playing scales. Third, we created a deterministic but unpredictable piece using the shift map. The shift map is $h(x) = 2x \pmod 1$. This map is chaotic on the unit interval [5], and its iterates fill the interval uniformly.

We first made a histogram of each voice, and divided the unit interval into twelve pieces, each having a length proportional to the probability of playing a given note. Then, using an arbitrary starting value, we iterated the shift map and chose the note corresponding to the interval dictated by the map. When we had the same number of notes as the original voice, we stopped. It should be noted that although the first two methods created music with exactly the same notes as the original, the third method created only a statistically similar distribution. Comparisons of the histograms for music created with the third method and the real music from which it was derived showed that the two were very close to having the same distribution.

>>> INSERT FIGURE 5 ABOUT HERE <<<

The results of dimension plots for the control data again for the Mozart piece are shown in Fig. 5. Notice that the real music is closer to completely predictable at small scales but that it is more nearly random at larger scales. This result suggests that the determinism in music is stronger over long time scales than over short time scales. That is to say, a short sequence of notes does not dictate the succeeding note, but a long series of notes does. Also, as mentioned earlier, there is no distinct plateau for the real music, whereas all three control types tend to have plateaus. The oscillations in the dimension curves appear to be an artifact of the non-uniform distribution of notes in each pitch class; they disappear when the distribution of notes in each pitch class is close to uniform. These oscillations are statistically significant, and their amplitude does not decrease with larger amounts of data.

L-system Music

Another type of control data was produced using algorithmic music, created with Lindenmayer systems (L-systems) by Stephanie Mason [6]. L-systems are recursive string rewriting systems, which were originally used to describe the patterns in plants and flowers [7]. Later, they were used to create graphical images and music [8]. This music is completely deterministic, but only partially predictable. When the dimension plot for such an L-system piece was compared to the dimension plot for a piece of real music (again the Mozart piece), the two were virtually indistinguishable, as seen in Fig. 6.

>>> INSERT FIGURE 6 ABOUT HERE <<<

Thus the proposed method as outlined cannot absolutely distinguish between music composed by a human and music composed by a computer. However, L-system music, while tonally similar to real music (although somewhat repetitive), is rhythmically very simple. A simultaneous analysis of duration and pitches might be able to differentiate between the two.

CONCLUSIONS AND FURTHER STUDY

We have described a first step in a novel technique for quantifying determinism in music. Preliminary results are enticing, especially given the numerous simplifications that were made.

Many avenues are open to further investigation. It would be desirable to have a better way to treat simultaneous voices. Not only would this allow more accuracy, by recognizing the harmonic components of music, but it would greatly expand the types of music that can be analyzed in this way. Incorporating octave information might be helpful in certain cases as well, especially if one can narrow the field of comparisons and set the pitch range appropriately. One example is to set the pitch range to 88 for comparison of piano compositions. As mentioned before, analysis of durations or simultaneous analysis of durations and pitches would also be helpful. Finally, since this is a general method for handling discrete data types, any discrete data can be analyzed. Examples include stock market prices, pieces of literature, sequences of DNA base pairs [2], and decimal or other base representations of numbers.

ACKNOWLEDGMENTS

We would like to thank the University of Wisconsin, Madison and the Hilldale Faculty/Undergraduate Research program for funding this project. We would also like to thank Jing Li from the University of Michigan for allowing us to use her scorefile handling code, Stephanie Mason from Virginia Polytechnic Institute and State University for the L-system scorefiles, and Prof. Stephen Dembski of the University of Wisconsin-Madison Music Department for correspondence and advice.

REFERENCES

1. M. F. Barnsley, *Fractals Everywhere*, Springer-Verlag, New York, 1988.

2. H. J. Jeffrey, Chaos Game Representation of Genetic Sequences, *Nucleic Acids Research* **18** (8), pp. 2163-2170, 1990.

3. H. J. Jeffrey, Chaos Game Visualization of Sequences, *Computers & Graphics* **16**(1), 25-33, 1992.

4. P. Grassberger, and I. Procaccia, Characterization of Strange Attractors, *Phys. Rev. Lett*. **50**, pp. 346-349, 1983.

5. R. Devaney, *An Introduction to Chaotic Dynamical Systems*, 2nd ed. (pp. 50), Springer-Verlag, New York, 1989.

6. S. Mason, *Lindenmayer Systems and Space-Filling Music,* Geometry Center Research Report GCG44, pp. 94-103, 1992.

7. A. Lindenmayer and P. Prusinkiewicz, *The Algorithmic Beauty of Plants*, Springer-Verlag, New York, 1990.

8. P. Prusinkiewicz, *Score Generation with L-Systems*, International Computer Music Conference 1986 Proceedings, pp. 455-457, 1986.

Direct reprint requests to:

Julien C. Sprott

Department of Physics

University of Wisconsin

1150 University Avenue

Madison, WI 53706

sprott@juno.physics.wisc.edu

FIGURE CAPTIONS

Figure 1: A written score of music cannot be modeled as a dynamical system without certain simplifications

Figure 2: Two examples of Iterated Function Systems; (a) with 4 possible input values; (b) using pitch classes as input values

Figure 3: Examples of the output of the IFS process; (a) with uniformly random notes; (b) Mozart's "Sonata in C"

Figure 4: The dimension curve for Mozart's "Sonata in C", typical of the dimension curves that are generated from real music

Figure 5: The dimension curves for Mozart's "Sonata in C" and three types of control music derived from it

Figure 6: The dimension curves for Mozart's "Sonata in C" and a piece of music created algorithmically from L-systems