My Way

Michael Gogins
http://michaelgogins.tumblr.com

Irreducible Productions
New York

6 March 2017

Outline

I talk about my way of doing algorithmic composition. Many references and examples herein are by hyperlink to the World Wide Web. Most of my pieces are made using Csound. Note: Some track titles got reversed with CDBaby.

1. My Goals

2. My Approach

3. My Results

Motivation

- I've loved classical music since childhood.
- I made up music in my head, and sometimes I wanted to compose, but I was afraid I had no talent — and music theory seemed *very* daunting!
- A 1978 Martin Gardner column showed me the Peano snowflake curve, and it occurred to me the horizontal lines could be punched as an interesting piano roll.
- When personal computers appeared in the 1980s, I started experimenting with algorithmic composition.
- But using fractals created problems that dragged me right back into music theory...

Peano Snowflake
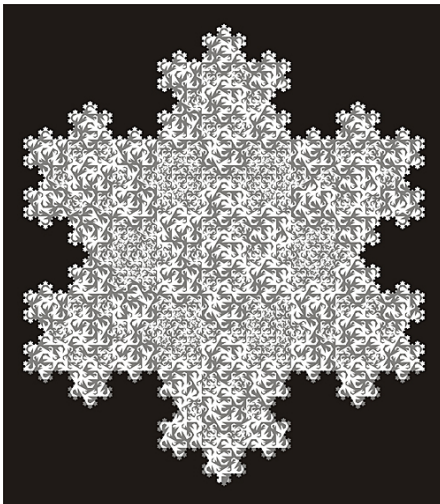
What Is Computer Music?

There are many definitions of computer music. The most obvious is, "Computer music is music that is made with a computer, just like piano music is music that is played on the piano." But that is not really a good definition, because computers today are used to record, mix, master, publish, and listen to classical music, rock music, jazz, and every other kind of music. Perhaps a better definition is, "Computer music is music that can *only* be made with a computer." In other words, it is music that is *idiomatic* to the computer. But what does that mean?

What Is a Computer Anyway?

- An algorithm is just a procedure that can be carried out step by step to produce a result, without needing to know what the steps *mean*. This is the technical definition. A cookie recipe is an algorithm, so is long division.
- For any single algorithm, a machine can be built to execute it. Such a machine is called a Turing machine, or *computer*. The idea is basic in logic, mathematics, even physics.
- A single Turing machine can be built to execute an algorithm that can simulate any and all other Turing machines. Such a machine is called a universal Turing machine, or *programmable computer*.
- You yourself, a laptop computer, or a smartphone are all physical realizations of universal Turing machines.
- But a computer can execute algorithms literally *billions of times faster* than you can.

The Musical Idioms of the Computer

Since human beings can also compute, the idioms of the computer build upon its blinding speed:

- **Analysis/Resynthesis** Computers can analyze digital audio, morph it or cross-synthesize it, resynthesize it, and combine it into collages — in ways that would simply be impossible by hand. Although this is actually the most common kind of computer music, I don't usually do this.
- **Algorithmic Composition** Computers can generate musical materials or scores in ways that would just take too long to do by hand.
  - *Corpus-based* The computer uses a database of existing materials to generate music by variation and recombination, even using machine learning.
  - *Mathematics-based* The computer uses fractals or other mathematical procedures to generate scores. This is what I usually do.

Why Do Algorithmic Composition?

- Algorithmic composition is the use of algorithms to compose music.
- But why bother? We already know how to compose.
- Because it's usually impossible to predict in detail what algorithms will do until you watch them run.
- Mathematicians call this *computational irreducibility*.
- Here's a simple example: Tell me what the 1st iteration of $y \Leftarrow \frac{1}{4} \, 0.9384932 \, y \, (1 - y)$ is. The 1,000th iteration.
- Therefore, you can use algorithms to create and control things you can't imagine.
- Therefore, *algorithms can amplify the imagination.*

Computational Irreducibility

- Computational irreducibility is what generates the stunning beauty of Nature: branchings, flows, clouds.

- Computational irreducibility arises from the iteration of nonlinear operations; in my little example, the nonlinearity comes from squaring *y*.

- Using irreducibility to generate art, whether representational or abstract, deeply illuminates Aristotle's thesis that art is the imitation of Nature.

- Another excellent example of irreducibility is Conway's Game of Life, a 2 dimensional cellular automaton: each iteration, each cell lives if exactly 3 of its nearest neighbors is alive, and otherwise dies.

Universal Computer/Constructor in Life
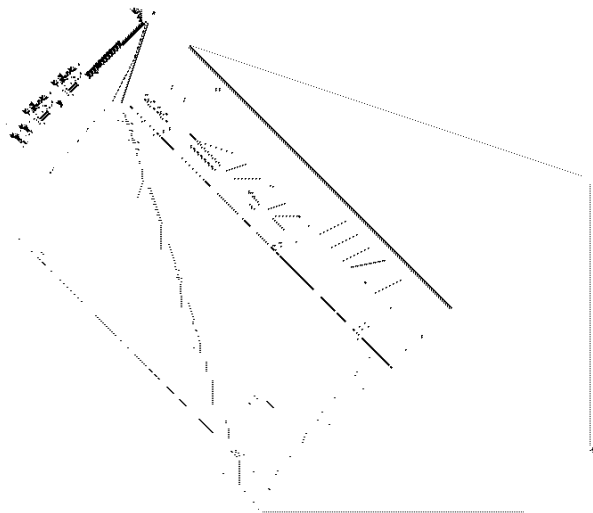
Algorithms I've Used I

- I started with chaotic dynamical systems, e.g. *Triptych*, 2004, chaos mapped onto predetermined chords.
- 1-dimensional cellular automata (rule 110 is also proved capable of universal computation).
- Then I played with Mandelbrot sets and Julia sets, and discovered a Mandelbrot-like set for music [4].
- Iterated function systems, e.g. *Chaotic Squares*, 1991, IFS mapped onto pitch $\times$ time.
- Lindenmayer systems, e.g. *Cloud Strata*, 1991, 0-L system mapped onto pitch $\times$ time.
- I worked with image generating software, and translated images to scores, e.g.
  06-f–2002-01-28–17-37-42.042.mml.9632.1

Algorithms I've Used II

- I used software like Kandid and Apophysis to *evolve* pieces, e.g. **_Sound Fractals_**, 2009, five evolved IFS mapped to time $\times$ frequency rendered via granular synthesis.
- I also used cell-based, accretive minimalist techniques, e.g. **_csound-2005-03-06–03.38.19.py_**
- There are many, many other useful algorithms that I *don't* use, many based on varying patterns from existing bodies of music, e.g. David Cope.

Tonality/Atonality

- I should point out that my pieces are not acousmatic, not abstractions of decontextualized sound.

- My music is made from notes and chords, and pays attention to pitch and time.

- Many of my earlier pieces were harmonically boring or hard to control, so recently I have added voice-leading transformations to my generative procedures.

- Such music *is not tonal* in that it is not necessarily constructed around key-establishing cadences.

- Such music *is tonal* in that it does use more or less consonant chords and common voice-leading moves.

Mathematical Music Theory

- Sometimes just mapping an irreducible algorithm onto pitch $\times$ time makes a good piece, usually *not*.
- So I decided that spaces score generators operate in should *already* have musical structure.
- In the mid 20th century composers and theorists gave "atonal theory" a basis in group theory.
- Now in the the late 20th and early 21st century, theorists have mathematized not only atonal theory, but also voice-leading ("neo-Riemannian theory").
- There's a lot of stuff out there, but I am using mostly the work of Dmitri Tymoczko on chord spaces [7] [3], and the work of Fiore and Satyendra on contextual transformations [5].

Algorithms I've Used – With Music Theory I

- **mkg-2007-01-20-c.py**, 2009. 0-L system mapped onto the Generalized Contextual Group [6].
- **Two Dualities**, 2010. 0-L system mapped onto the Generalized Contextual Group [6].
- **Blue Leaves 4e**, 2012. Recurrent IFS mapped onto pitch $\times$ time, filtered by generated chords.
- **Scrims**, 2016. Visual music for live performance: hopalong fractal variations in WebGL, sampled and subjected to chord transformations.

Production

- Each piece I write is a custom computer program. I have in the past used Basic, Fortran, Visual Basic, Pascal, C++, Java, Python, and Lua.
- I now mostly use HTML5, i.e. JavaScript with HTML, with Csound embedded. I use Csound for Android and csound.node.
- Each piece embeds Csound and a complete Csound orchestra. I aim for a finished level of sound design and mastering.
- In my text editor, I tinker with a few lines of code, press a key, and get a rendered piece.
- This approach would work just as well with SuperCollider, RTcmix, PD, or Max.

Publication

- My music is not popular music. I don't have anything against such music; indeed, I listen to it, and learn from it.
- My music is art music for undistracted listening in the tradition of Bach, Xenakis, etc.
- I also do some visual music, where I generate animated video along with the music.
- I play my pieces in festivals and conferences of computer music and electroacoustic music, such as the NYCEMF.
- I publish some pieces on CDBaby.
- I publish other pieces on SoundCloud or YouTube.
- All my pieces are registered with the U.S. Copyright Office, ASCAP, and AdRev so if somebody listens to my music online, I *will* get paid. Of course the pay is laughable!

References I

📕 Michael Gogins, blog.

📕 Michael Gogins. "Computer Music Resources."

📕 Clifton Callender, Ian Quinn, and Dmitri Tymoczko. "Generalized voice-leading spaces." *Science*, 320:346–348, 2008.

📕 Michael Gogins. "How I Became Obsessed with Finding a Mandelbrot Set for Sounds," **News of Music 13**:129-139.

📕 T.M. Fiore and R. Satyendra. "Generalized Contextual Groups." *Music Theory Online*, 11(3), 2005.

References II

📕 Michael Gogins. "Score generation in voice-leading and chord spaces." In Georg Essl and Ichiro Fujinaga, editors, *Proceedings of the 2006 International Computer Music Conference*, San Francisco, California, 2006. International Computer Music Association.

📕 Dmitri Tymoczko. "The Geometry of Musical Chords." *Science*, 313:72–74, 2006.