

# Compositional experiments with concatenating distinct waveform periods while changing their structural properties.

Arun Chandra  
Computer Music Project  
School of Music  
University of Illinois  
Urbana, IL 61801  
arunc@evergreen.edu

April 18, 1998

## Abstract

*wigout* is a sound-synthesis program, written in C and running under Unix and 32-bit Intel systems. The premise of the program is to allow the composer to compose the waveform with which she composes. Thus, sound is not a building-block with which one composes, but the subject matter of composition.

The composer defines a *waveform state*, consisting of an arbitrary number of *segments*. Each *segment* is similar to (but not identical with) 1) a sine wave; 2) a square wave; 3) a triangle wave; or 4) a sawtooth wave. The composer stipulates the duration for which the sound is to last, and then the *waveform state* (which is on the order of a few milliseconds long) is iterated until the desired duration is reached. Upon each iteration, each segment changes itself by a specified amount.

The resulting sound is the result of many independent changes in the waveform's segments.

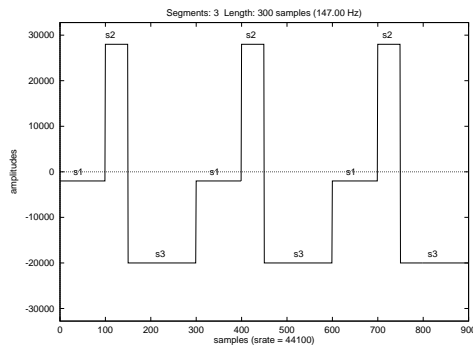
Up till now, five compositions have been written using *wigout*, for tape alone, and for tape and performers.

I would like the completely determined chaos of *wigout* to be understood in opposition to two powerful lines of thought: 1) that of religious or merchantile determinism, where everthing is known and securely predictable, and freedom is lost; and 2) that of religious or postmodern relativism, where nothing is knowable, all alternatives are equal, and freedom is lost. Opposition to these to is the philosophical base of *wigout*: a system in which although everthing is determined, the outcome is difficult to predict.

## Wigout

Wigout is a program for time-domain synthesis. In frequency-domain synthesis, the composer specifies the frequency and amplitudes she wants, and gets that result. In time-domain synthesis, one specifies the structure of a waveform, and how it changes over time. The constructed waveform then, creates the frequency and amplitude changes.

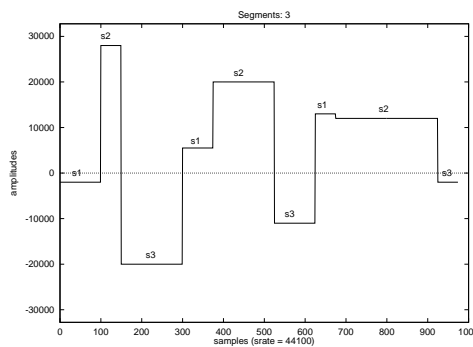
Wigout defines a waveform as sequence of *segments*:



In this slide, there are three segments, and that sequence is iterated 3 times. This sequence has a duration of about 7 thousandths of a second.

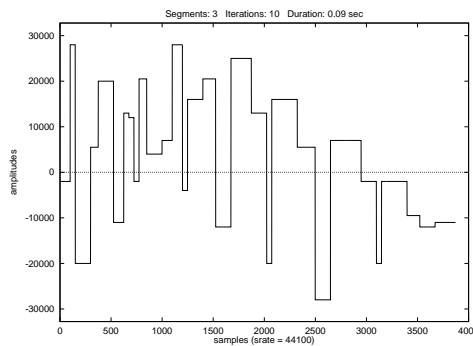
Each segment has two variables: a duration (measured in samples) and an amplitude (between  $\pm 32767$ ). In this example, on each iteration the segments' amplitude and duration remain the same.

If both amplitude and duration were changing, the three iterations might look like this:

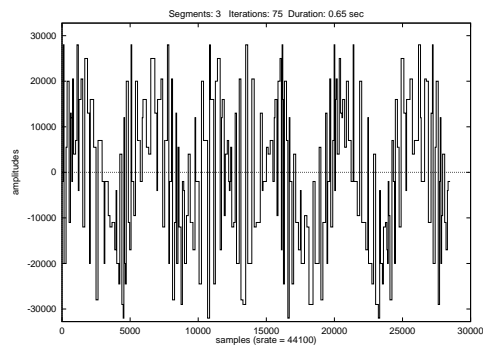


This is the fundamental idea of wigout: on each iteration, every segment changes its amplitude and its duration by a specified increment. When either variable reaches its minimum or maximum limit (which can be specified), that variable reverses its direction of change: If it was increasing, it starts decreasing; if decreasing, it starts increasing.

After 10 iterations (about 1/10 of a second), the waveform looks like this:



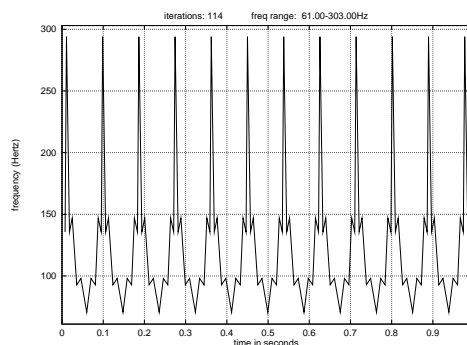
After 100 iterations (about 6/10 of a second), it looks like this:



Here's what 10 seconds of this transformation sound like:

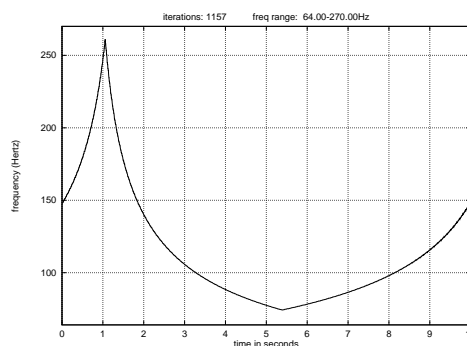
Sound example 1: 10 seconds

And here's a plot of how the frequencies are changing during the first second. The other slides I showed you were slides of the waveform. This is a slide of the frequency changes.



The duration of a segment determines its frequency. If I reduce the duration increment, I slow down the frequency's rate of change.

The frequency plot then looks like this:



And it sounds like this:

Sound example 2: 10 seconds

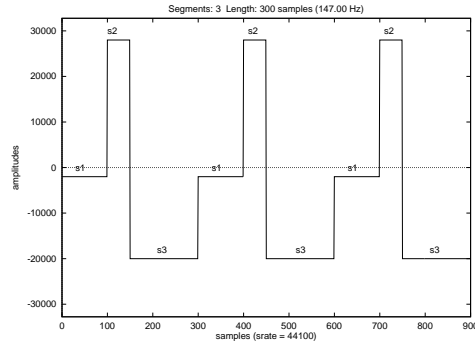
If, in addition to reducing the duration increment, I also reduce down the amplitude increment, the result sounds like this:

### Sound example 3: 10 seconds

The changes in amplitude are now a kind of intermittent vibrato.  
Reducing the amplitude increment even further sounds like this:

### Sound example 4: 10 seconds

## Reiterate



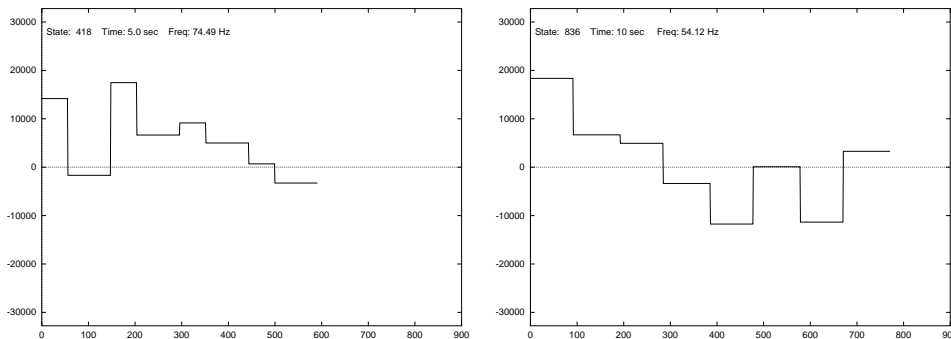
To reiterate what I've said:

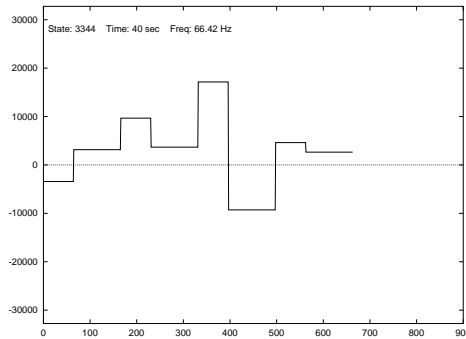
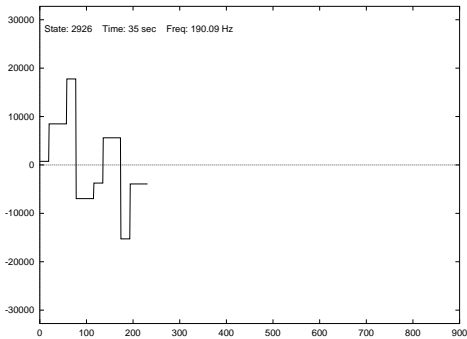
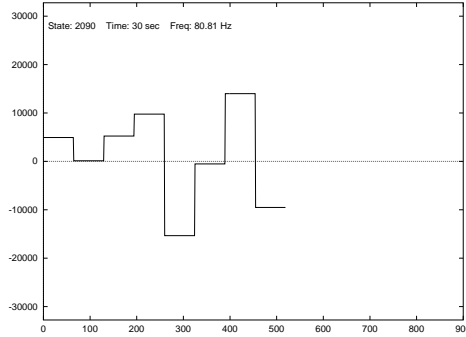
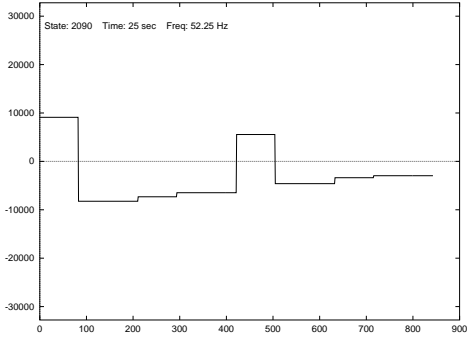
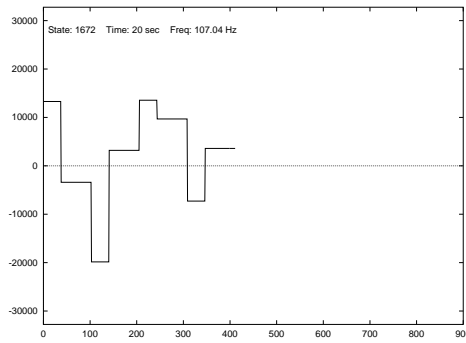
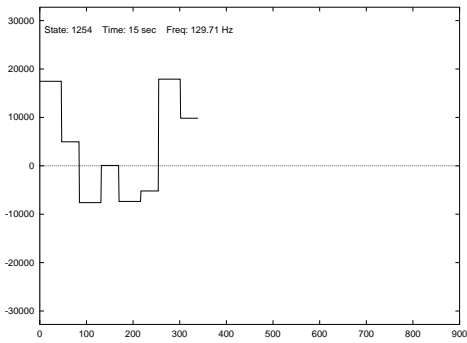
1. A waveform is defined by specifying a sequence of *segments*.
2. A segment has an amplitude and a duration.
3. The sequence is iterated, and on each iteration, every segment changes its amplitude and duration by a specified increment.
4. When the amplitude and duration reach their limits, they reverse the direction of their change.

Here's an extended example (80 seconds long). This is a sequence with eight segments, and both the duration and the amplitude of every segment are changing at unique increments.

This is a plot of 8 states of the transformation, taken 5 seconds apart over the first 40 seconds.

### Sound example 5: 80 seconds





## Twiggles

All the above examples are essentially square waves, whose duration and amplitude are independently varying.

Wigout allows for the building of segments that are not “square wave-like”, but rather similar to triangle or sine waves.

Square-wave-like segments are named *wiggles*. Triangle-wave-like segments are named *twiggles*.

Here’s a sequence of 6 segments, and each segment is a twiggle. [SHOW THE TWIGGLE SLIDE]

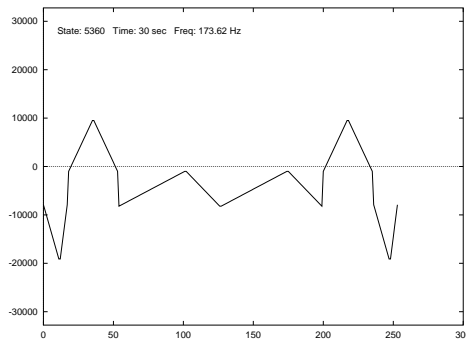
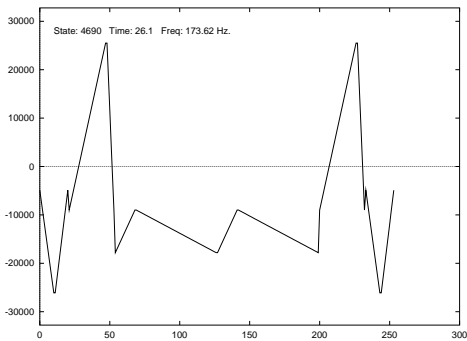
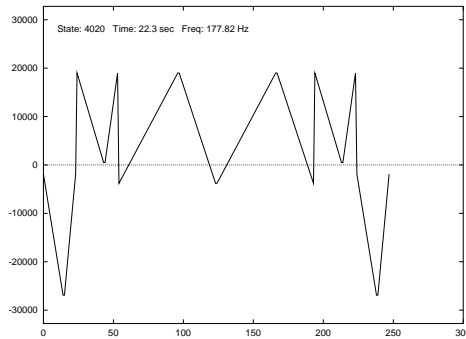
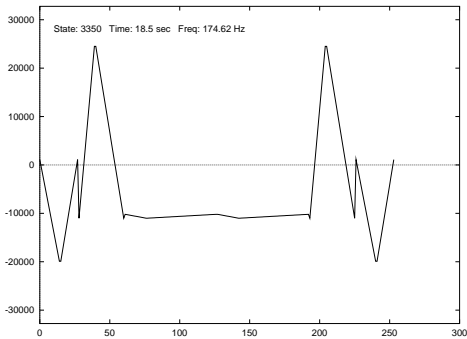
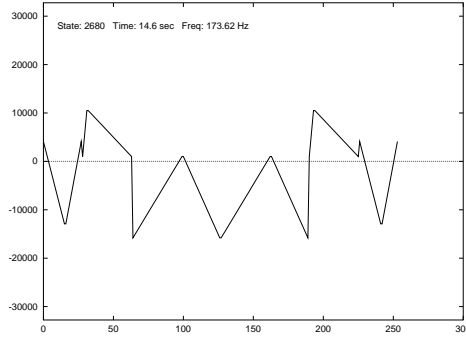
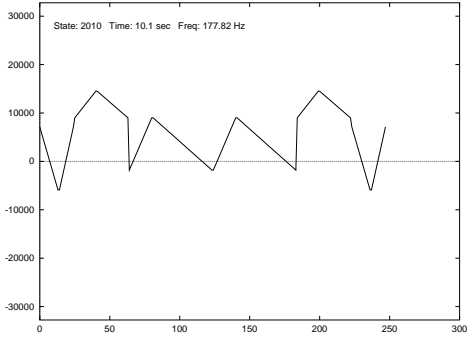
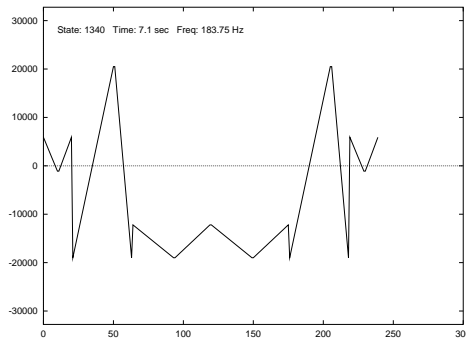
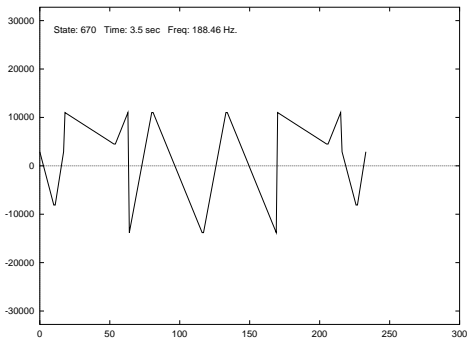
Like a wiggle, a twiggle has a duration and an amplitude that are independently varying. In addition, a twiggle also has a triangular peak, whose amplitude and location can also vary. A Twiggle thus has four independent variables: duration, amplitude, peak amplitude, and peak location.

This slide shows the changing waveform over 30 seconds, and the plots are 3.5 seconds apart.

Wigout allows for the arbitrary combination of segments, so, in this example, segments 1 and 6 are identical, 2 and 5 are identical, and 3 and 4 are identical.

This transformation sounds like this:

Sound example 6: 30 seconds



## Ciggles

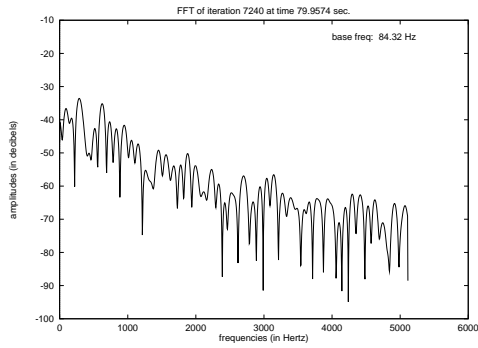
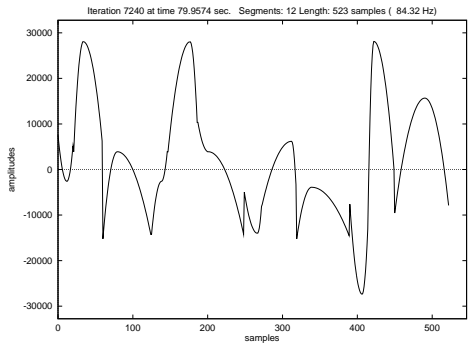
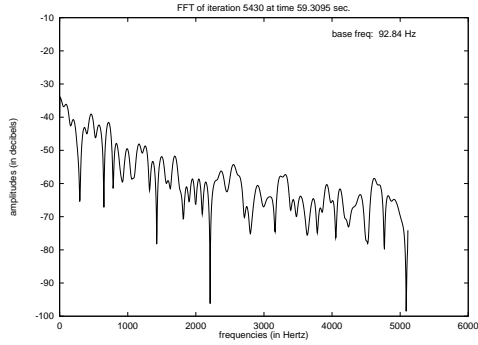
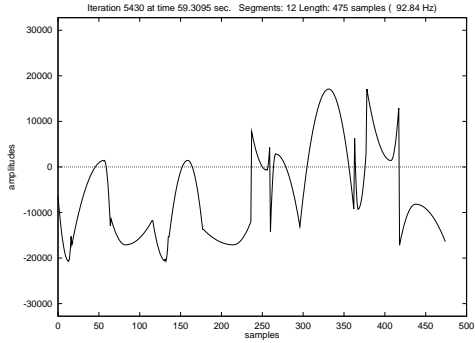
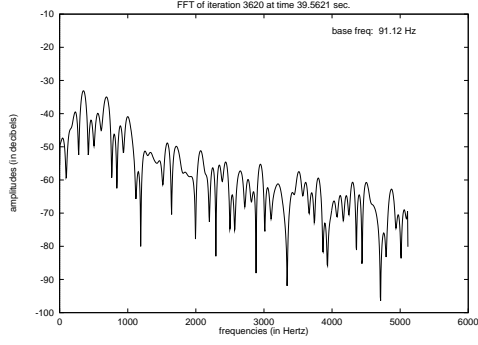
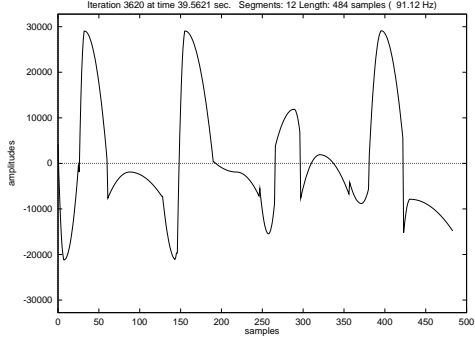
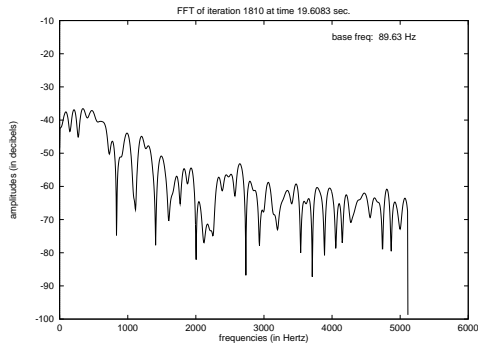
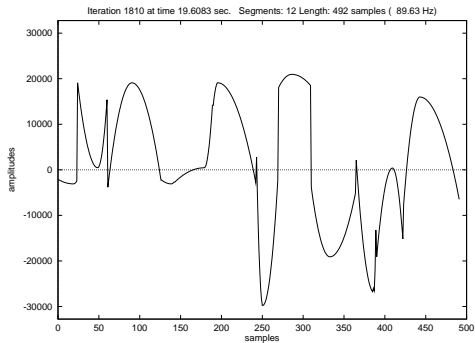
In addition to wiggles and twiggles, there are also ciggles: or curved twiggles.

Here's an example of a sequence with twelve distinct segments, where each segment is a ciggle.

These plots are from an 80-second transformation; the plots are about 20 seconds apart. Next to each state is its FFT. You can see that, indeed, the timbres do change.

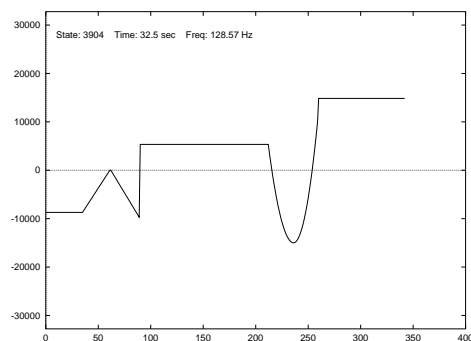
This 80-second example sounds like this. Please note the changing harmonic envelopes.

Sound example 7: 80 seconds



## Multiple shapes in a single waveform

Wigout allows for multiple shapes to be combined in one waveform, so arbitrary combinations of wiggles, twiggles, and ciggles can be built. One “multiple-shape” waveform could look like this:



Wigout also allows for multiple simultaneous instances of waveforms, so that overlaps and placement within a stereo field are possible.

The last example is a composition I wrote for trombone, recorded voice, and tape. The entire composition lasts 12 minutes, and I'll play the first three minutes of the tape part.

## Biography

Arun Chandra, born 1954 in Allahbad, India, raised in the United States, completed his DMA in Music Composition at the University of Illinois (Urbana/Champaign), where his primary teachers were Herbert Brün, Alexander Ringer, and Ben Johnston. (Others who have had a strong influence on his compositions and thinking include Kenneth Gaburo, Heinz von Forester, and Humberto Maturana.) He studied guitar with Alvaro Company in Florence, Italy and with Allain LeBellac in Fountainbleau, France, and studied conducting with Michael Gielen at the Mozarteum in Salzburg. From 1980-92 he toured extensively in the United States and Europe with the Performers' Workshop Ensemble, and shared with them a 6-month residency at the "Hochschule für Musik" in Kassel, West Germany. For three years he worked at Wolfram Research, Inc., where he designed and wrote the *Music* and *Audio* packages for Mathematica 2.x. He was Associate Professor of Music Composition at the Institute for Applied Arts, Hsinchu, Taiwan, where he directed the computer music studio, and conducted the orchestra, in addition to teaching composition and theory. His compositions have been performed at numerous international and national conferences and festivals, as well as on the radio in the U.S., Europe, Australia, and Japan. He has given numerous presentations of his research and compositions at conferences and symposia in Europe, the United States, and Asia. He lives in Olympia, Washington, with his partner Lori Blewett and their two children. He teaches students in music and music composition at The Evergreen State College, and is music director and conductor of the Olympia Chamber Orchestra.