

The Synthesis Algorithms Used by TRIKTRAKS and WIGOUT

Arun Chandra

arunc@evergreen.edu

Contents

1	Introduction	2
2	How does a computer make sound?	3
3	What is a sample?	5
4	What is a waveform?	6
5	How do Wigout and TrikTraks compose waveforms?	7
6	How does a computer produce a frequency?	9
7	How does a computer produce loudness?	10
8	What else do I need to know about the basis of TrikTraks and Wigout?	11
9	Converting Pitch and Frequency to Samples	12
10	Converting Dynamics and Decibels to Amplitudes	13

1 Introduction

This is a brief introduction to the premises that underlie WIGOUT and TRIKTRAKS.

The reader need not be a mathematician to understand this paper. If you know arithmetic, and basic high-school algebra, you will be able to follow the discussion.

Both WIGOUT and TRIKTRAKS are *time-domain synthesis* programs. That means they manipulate and synthesize sounds by way of the duration and amplitude of the waveforms.

These programs take up on the ideas first explored by Herbert Brün, with his sound synthesis program SAWDUST, designed and implemented by Gary Grossman, and enhanced by Jody Kravitz and Keith Johnson. In particular, the idea of treating the square wave as the fundamental building block for sound synthesis comes from SAWDUST.

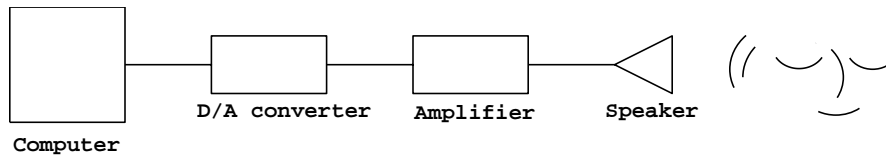
I'll start with an explanation of how a computer makes sound, and then how one can, with the help of a program, compose the waveforms the computer creates.

Above all, the reader is invited to enter into this program in a spirit of play, despite all the “imposing mathematics” and whatnot. The pleasure of the program, for me and for a few other people at least, is the pleasure in the unpredictability of its output, despite a totally determined input. (If you can enjoy that last thought, then you'll have a good time with this program.)

As Gloucester says of his bastard son Edmund at the beginning of *King Lear* “Though the knave came something saucily into the world before he was sent for, yet was his mother fair, there was good sport at his making, and the whoreson must be acknowledged.” Something of that applies to my relationship to these programs as well . . .

2 How does a computer make sound?

Here's a description of how a computer makes sound, starting from the ear, and going back to the computer.



The ear detects changes of pressure in the air. If those changes occur between 20 and 20,000 times per second, the brain, attached to the ear, will interpret those changes as sound.

Where do those changes of air pressure come from?

From a speaker, that translates a voltage into the movement of a speaker cone. Let's say the voltage can be between ± 1 volts. When the speaker receives a voltage of +1, the cone is pushed all the way out. When it receives a voltage of -1, the cone is pulled all the way in. Changes of voltage between +1 and -1 cause the speaker to take up intermediary positions, somewhere between all the way out and all the way in.

This movement of the speaker creates the changes in the air, which are noticed by the ear, and understood as sound.

Where does the electrical current come from that drives the speaker?

From an amplifier, that amplifies an incoming signal so that it can be used by a speaker.

Where does the amplifier get its incoming signal from?

From a Digital to Analog converter (D/A converter), that converts numbers into voltages. (The amplifier could, of course, get its incoming signal from a CD player, or a radio, or other output device.)

Where does the D/A converter get its numbers from?

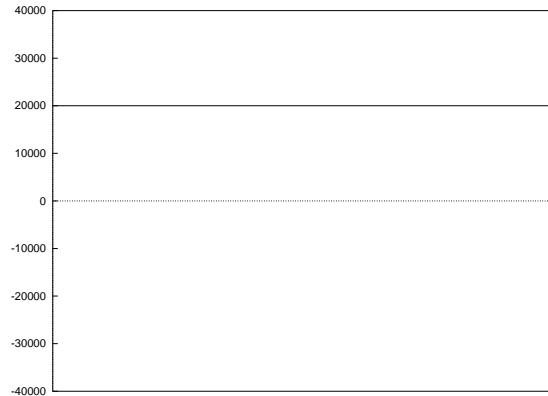
From a computer, that feeds the converter a series of numbers at a specified rate. The *range* of numbers a converter can handle is its *sample width* (usually 16-bits, nowadays). The *rate* at which the converter can translate numbers into voltages is its *sampling rate*.

If the converter is a 16-bit converter, that means it can accept numbers between +32767 and -32768. The converter interprets those numbers at a particular speed, called its *sampling rate*. A *sampling rate* is a division of one second into a stipulated number of equal parts. The standard CD sampling rate is 44100 samples per second. A converter converts 44100 numbers, each one between +32767 and -32768, every second, for one channel of sound. (For stereo, the converter has to convert twice as many numbers, first left, then right, and so on.)

How does a computer know which numbers to send to the D/A converter?

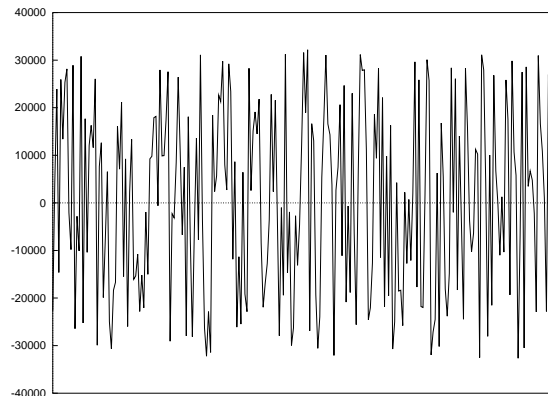
A program that controls the computer tells it what to send to the converter.

If the program keeps sending the same number to the converter (say, 20,000), the result would look like this:



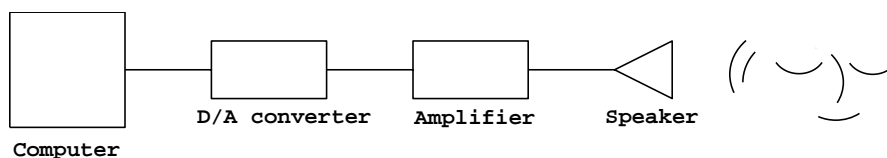
One would hear only an initial and final click, and the rest would be silence.
In the absence of change, sound cannot exist.

If the program randomly picks numbers between ± 32767 , the result might look like this:



If the above sequence of numbers were sent to a D/A converter, the result would be called “white noise” (a sound where *all* frequencies are of the same magnitude).

Recapitulate



A program tells the computer to make numbers, which are sent to a D/A converter, which converts them to voltages, which are sent to an amplifier, which amplifies them and sends them to a speaker, which transforms the voltages into changes of air pressure, which are detected by the ear, which get sent to the brain, which understands them as sound.

3 What is a sample?

A sample is a number sent to the D/A converter. A sample has a *magnitude* and a fixed *duration*.

- The magnitude on a 16-bit converter is ± 32767 .
- The duration is the inverse of the converter's sampling rate ($1/sr$).

$$1 \text{ sample} = 1/44100 = 0.00002 \text{ second}$$

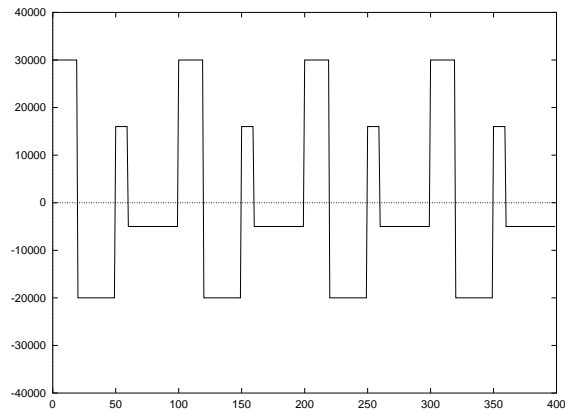
At a sampling rate of 44100 samples per second, each sample has a duration of 0.00002 second.

(One second is a long time for a computer. That's why we divide it into samples. A sampling rate of 10 samples per second would divide the second into tenths, with each sample lasting 0.1 second.)

4 What is a waveform?

A waveform is an identifiable sequence of samples.

If the computer program picked four numbers between ± 32767 (30000, -20000, 16000, and -5000 in the example below), and repeated that sequence, the result might look like this:



Assuming a sampling rate of 44100 samples per second:

- +30000 is repeated for 20 samples (0.0004 sec.)
- -20000 is repeated for 30 samples (0.0006 sec.)
- +16000 is repeated for 10 samples (0.0002 sec.)
- -5000 is repeated for for 40 samples (0.0008 sec.)

A “waveform” is a single iteration of this sequence (100 samples long, 0.002 seconds long). Repetition of this sequence produces a sound.

5 How do Wigout and TrikTraks compose waveforms?

Both WIGOUT and TRIKTRAKS use *elements* that are joined together to form *states*.

An element has a duration and an amplitude.

Its duration is specified in samples, and its amplitude is a number between +32767 and -32768.

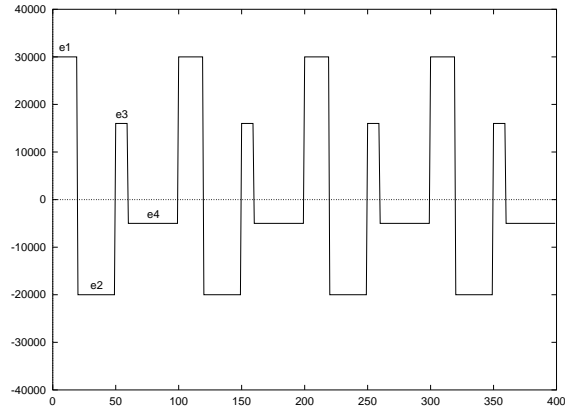
The waveform described above, has four elements:

- element 1: 20 samples, +30000 amplitude
- element 2: 30 samples, -20000 amplitude
- element 3: 10 samples, +16000 amplitude
- element 4: 40 samples, -5000 amplitude

After defining your elements, you define a state that uses your elements. The state defined above is:

state 1: element 1, element 2, element 3, element 4

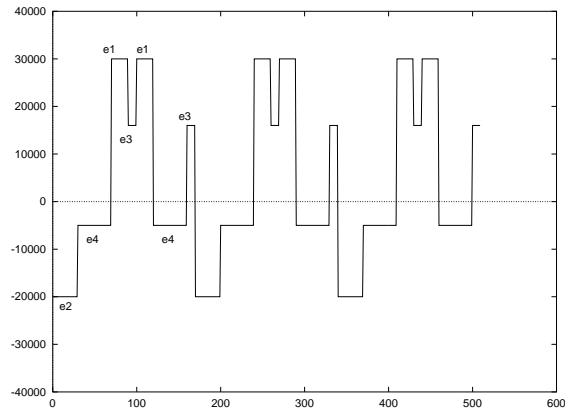
And it looks like this:



The same four elements could be used to define a different state:

state 2: e2, e4, e1, e3, e1, e4, e3

And the waveform would look like this:



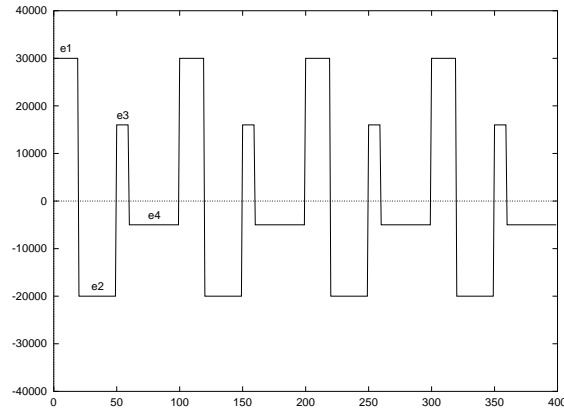
These two steps are the fundamentals for using both WIGOUT and TRIKTRAKS:

1. Define a collection of elements.
2. From those elements, define a collection of states.

The rest of this introduction describes how the information contained in elements and states is converted into frequency and loudness.

6 How does a computer produce a frequency?

A computer produces a frequency by repeating a waveform.



If you look carefully at the plot above, you will see that the waveform repeats every 100 samples. There are four complete iterations of the waveform in the above plot.

A waveform's duration determines the fundamental frequency of the sound.

At the sampling rate of 44100 samples per second, this waveform's fundamental frequency is:

$$44100 / 100 = 441 \text{ hertz}$$

So, it is just over 440 hertz, which is a standard, tuning pitch "A".

The general formula for converting waveform durations into frequencies is:

$$\text{frequency} = \text{sampling_rate} / \text{waveform_duration_in_samples}$$

If the sampling rate were changed to 22050 samples per second, and the very same waveform were sent to the converter, the frequency would be different:

$$22050 / 100 = 220.5 \text{ hertz}$$

Exactly one octave lower than when the sampling rate was 44100 samples per second.

7 How does a computer produce loudness?

A computer does not directly produce loudness. (The amplifier does that.) What the computer produces is differences in amplitude, or the *maximum displacement* within a waveform.

As stated above, a 16-bit D/A converter accepts numbers between +32767 and -32768, and converts them into a continuously varying voltage. Thus, the maximum displacement (the difference between +32767 and -32768) is:

$$32767 - -32768 = 65536$$

Converting that number into decibels (a common way to measure magnitude):

$$20 * \log_{10}(\text{max_displacement}) = \text{decibels}$$

$$20 * \log_{10}(65536) = 96.32 \text{ dB}$$

So, the greatest magnitude that can be produced by a 16-bit converter is about 96 dB. By way of metaphor, the sound of people quietly talking in a room is about 60 dB, and the sound of an airplane taking off is about 120 dB.

(By the way, for the purposes of this discussion, it is not necessary that the reader understand the mathematical implications of logarithms. Rather, the salient point is to understand that this is a way of measuring the loudness of a waveform created by a computer.)

In the square wave example above, three numbers were being sent to the D/A converter: 30000, -20000, 16000, and -5000. The maximum displacement here is between +30000 and -20000, which is 50000. Converting this number into decibels:

$$20 * \log_{10}(50000) = 93.97 \text{ dB}$$

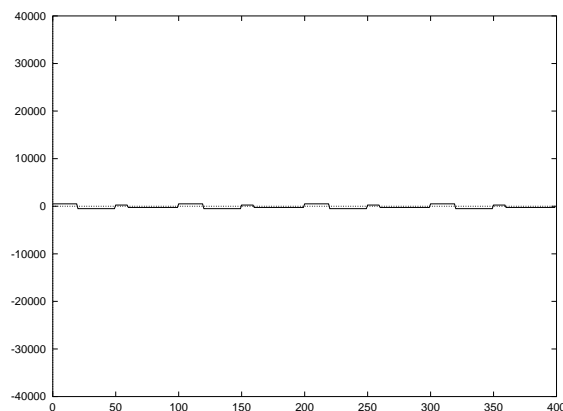
or almost 94 dB, close to the maximum of what a 16-bit converter can produce.

If we wanted to produce a waveform whose maximum amplitude was 60 dB, then we would use the following formula:

$$10^{(\text{dB}/20)} = \text{max_displacement}$$

$$10^{(60/20)} = 1000$$

This means that the maximum displacement we need is 1000. So, a 60 dB waveform might look like this:



The number +500 is output for 20 samples, then -500 for 30 samples, +250 for 10 samples, and -250 for 40 samples. The *maximum displacement* is between +500 and -500, which is 1000.

The fundamental frequency of this waveform is the same as the waveform above (441 hertz). Only the loudness is now 60 dB rather than the 94 dB in the earlier example.

8 What else do I need to know about the basis of TrikTraks and Wigout?

Not much. Remember that the frequency is the inverse of the waveform duration:

$$frequency = sampling_rate / waveform_duration$$

$$waveform_duration = sampling_rate / frequency$$

And also that the volume is

$$decibels = 20 * \log_{10}(maximum_displacement)$$

$$maximum_displacement = 10^{decibels/20}$$

So, if you wanted to create the frequency of 660 hertz (the E above A-440) and your sampling rate was 44100

$$66.81 = 44100 / 660$$

You would need a waveform duration of 67 samples (rounded up). If you wanted that waveform to have a loudness of 72 dB:

$$3981.07 = 10^{72/20}$$

Its maximum displacement would have to be 3981.

9 Converting Pitch and Frequency to Samples

name	Hertz	Sampling Rates				name	Hertz	Sampling Rates			
		11025	22050	44100	48000			11025	22050	44100	48000
a0	27.500	401	802	1604	1745	a5	880.000	13	25	50	55
a#0	29.135	378	757	1514	1647	a#5	932.328	12	24	47	51
b0	30.868	357	714	1429	1555	b5	987.767	11	22	45	49
c0	32.703	337	674	1348	1468	c5	1046.502	11	21	42	46
c#0	34.648	318	636	1273	1385	c#5	1108.731	10	20	40	43
d0	36.708	300	601	1201	1308	d5	1174.659	9	19	38	41
d#0	38.891	283	567	1134	1234	d#5	1244.508	9	18	35	39
e0	41.203	268	535	1070	1165	e5	1318.510	8	17	33	36
f0	43.654	253	505	1010	1100	f5	1396.913	8	16	32	34
f#0	46.249	238	477	954	1038	f#5	1479.978	7	15	30	32
g0	48.999	225	450	900	980	g5	1567.982	7	14	28	31
g#0	51.913	212	425	849	925	g#5	1661.219	7	13	27	29
a1	55.000	200	401	802	873	a6	1760.000	6	13	25	27
a#1	58.270	189	378	757	824	a#6	1864.655	6	12	24	26
b1	61.735	179	357	714	778	b6	1975.533	6	11	22	24
c1	65.406	169	337	674	734	c6	2093.005	5	11	21	23
c#1	69.296	159	318	636	693	c#6	2217.461	5	10	20	22
d1	73.416	150	300	601	654	d6	2349.318	5	9	19	20
d#1	77.782	142	283	567	617	d#6	2489.016	4	9	18	19
e1	82.407	134	268	535	582	e6	2637.021	4	8	17	18
f1	87.307	126	253	505	550	f6	2793.826	4	8	16	17
f#1	92.499	119	238	477	519	f#6	2959.956	4	7	15	16
g1	97.999	113	225	450	490	g6	3135.964	4	7	14	15
g#1	103.826	106	212	425	462	g#6	3322.438	3	7	13	14
a2	110.000	100	200	401	436	a7	3520.000	3	6	13	14
a#2	116.541	95	189	378	412	a#7	3729.310	3	6	12	13
b2	123.471	89	179	357	389	b7	3951.066	3	6	11	12
c2	130.813	84	169	337	367	c7	4186.009	3	5	11	11
c#2	138.591	80	159	318	346	c#7	4434.922	2	5	10	11
d2	146.832	75	150	300	327	d7	4698.637	2	5	9	10
d#2	155.564	71	142	283	309	d#7	4978.032	2	4	9	10
e2	164.814	67	134	268	291	e7	5274.042	2	4	8	9
f2	174.614	63	126	253	275	f7	5587.652		4	8	9
f#2	184.997	60	119	238	259	f#7	5919.912		4	7	8
g2	195.998	56	113	225	245	g7	6271.928		4	7	8
g#2	207.652	53	106	212	231	g#7	6644.876		3	7	7
a3	220.000	50	100	200	218	a8	7040.000		3	6	7
a#3	233.082	47	95	189	206	a#8	7458.620		3	6	6
b3	246.942	45	89	179	194	b8	7902.133		3	6	6
c3	261.626	42	84	169	183	c8	8372.019		3	5	6
c#3	277.183	40	80	159	173	c#8	8869.845		2	5	5
d3	293.665	38	75	150	163	d8	9397.273		2	5	5
d#3	311.127	35	71	142	154	d#8	9956.064		2	4	5
e3	329.628	33	67	134	146	e8	10548.083		2	4	5
f3	349.228	32	63	126	137	f8	11175.305			4	4
f#3	369.994	30	60	119	130	f#8	11839.823			4	4
g3	391.995	28	56	113	122	g8	12543.855			4	4
g#3	415.305	27	53	106	116	g#8	13289.752			3	4
a4	440.000	25	50	100	109						
a#4	466.164	24	47	95	103						
b4	493.883	22	45	89	97						
c4	523.251	21	42	84	92						
c#4	554.365	20	40	80	87						
d4	587.330	19	38	75	82						
d#4	622.254	18	35	71	77						
e4	659.255	17	33	67	73						
f4	698.457	16	32	63	69						
f#4	739.989	15	30	60	65						
g4	783.991	14	28	56	61						
g#4	830.609	13	27	53	58						

10 Converting Dynamics and Decibels to Amplitudes

This chart shows how to convert decibels into linear amplitudes, with approximate musical dynamics.

dynamic	dB	linear
pppp	60.00	+500
ppp	64.04	+796
pp	68.07	+1267
p	72.11	+2016
mp	76.15	+3208
mf	80.18	+5107
f	84.22	+8127
ff	88.26	+12936
fff	92.29	+20588
ffff	96.33	+32768