Creating and Using a Wavetable in MATLAB

Introduction

There are two basic ways to generate digital sinusoidal waveforms. One is to evaluate a sinusoidal function over time. This involves dynamic computation of the sine function, along with whatever mathematical operations are used to modulate the waveform further. A second way to generate a waveform is by means "table-lookup" -- that is, reading stored values out of memory. The basis of this method is a table-lookup oscillator.

A table-lookup oscillator is a set of look-up wavetables stored in contiguous memory locations. Each wavetable contains a list of sample values constituting one cycle of a sinusoidal wave, as illustrated in Figure 1. Multiple wavetables are stored so that waveforms of a wide range of frequencies can be generated.

With a table-lookup oscillator, a waveform is created by starting at an index in the table and outputting consecutive values, cycling back to the beginning of the table as necessary. Let's experiment with this to see how create a wavetable and output the desired frequency from it. With a table of \( N \) samples representing one cycle of a waveform and an assumed sampling rate of \( r \) samples/s, you can generate a fundamental frequency of \( r/N \) Hz simply by reading the values out of the table at the sampling rate. An example of a wavetable is given in Figure 1 for a frequency of 375 Hz and a sampling rate of 48000 Hz.
The wavetable is actually the list of values on the left, stored in memory. The figure simply illustrates the fundamental waveform that these values correspond to.

A wavetable such as the one shown can be used as the basis of more than its fundamental frequency. By varying the rate at which values are output, harmonic frequencies can be produced. For example, reading every other value results in a waveform of twice the frequency of the fundamental. Reading each value twice (or adding one in between each two by interpolation) results in a waveform of half the frequency of the fundamental.

The phase of the waveform can also be varied by starting at an offset from the beginning of the wavetable. To start at a phase offset of $p\pi$ radians, you would start reading at index $\frac{pN}{2}$. For example, to start at an offset of $\pi/2$ in the wavetable of Error! Reference source not found., you would start at index $\frac{1+128}{2} = 32$.

To generate a waveform that is not a harmonic of the fundamental frequency, it’s necessary to move from one index to another in the table by adding the correct increment. This increment $i$ depends on the desired frequency $f$, the table length $N$, and the sampling rate $r$, defined by $i = \frac{f \cdot N}{r}$. For example, to generate a waveform with frequency 750 Hz using the wavetable of Error! Reference source not found. and assuming a sampling rate of 48000 Hz, you would need an increment of $i = \frac{750 \cdot 128}{48000} = 2$.

We’ve chosen an example where the increment is an integer, which is good because the indexes into the table have to be integers. What if you wanted a frequency of 390 Hz? Then the increment would be $i = \frac{390 \cdot 128}{48000} = 1.04$, which is not an integer. In cases where the increment is not an integer, interpolation must be used. For example, if you want to go an increment of 1.04 from index 1, that would take you to index 2.04. Assuming that our wavetable is called $table$, you want a value equal to $table[2] + 0.04 \cdot (table[3] - table[2])$. This is a simple way to do interpolation. More complex, curve-shaping forms are also possible.

**The Assignment**

Your assignment is to write the following MATLAB functions:

\[ table, N \] = wavetable(f, r), which generates a wavetable called $table$ that contains one cycle of a sinusoidal waveform of frequency $f$ with a sampling rate of $r$. The number of samples in the table, $N$, is also returned.

\[ [i, waveform] = makeSoundWithWavetable( table, r, freqDesired, s) \] sends in a wavetable $table$, its sampling rate $r$, the desired frequency of a sound to generate $freqDesired$, and the desired length of the sound in seconds $s$. The function plays the sound of the desired waveform within the function and sends back the vector of values in $waveform$ and the increment used to get these values $i$. Use the simple interpolation method explained above. Have your function plot one cycle of $waveform$. 

This material is based on work supported by the National Science Foundation under CCLI Grant DUE 0717743, Jennifer Burg PI, Jason Romney, Co-PI.