Waveshaping Synthesis

- In waveshaping, it is possible to change the spectrum with the amplitude of the sound (i.e. changing the time-domain waveform by a controlled distortion of the amplitude).

- Since this is also a characteristic of acoustic instruments, waveshaping has been used effectively for synthesizing traditional musical instruments, and in particular, brass tones.

- Like FM, waveshaping synthesis enables us to vary the bandwidth and spectrum of a tone in a way that is more computationally efficient than additive synthesis.

- Also like FM, waveshaping provides a continuous control of the spectrum over time by means of an index.

- Unlike FM, waveshaping allows you to create a band-limited spectrum with a specified maximum harmonic number (i.e. making it easier to prevent aliasing!).

Waveshaper

- In a simple waveshaping instrument, an input signal \( x(t) \) is passed through a box containing a waveshaping function or transfer function, also known as a waveshaper, \( w(x) \).

\[
x(t) \rightarrow w(x) \rightarrow y(t)
\]

Figure 1: A simple waveshaping instrument with a waveshaping transfer function \( w(x) \).

- The transfer function \( w(x) \) is typically nonlinear, and alters the shape of the input \( x(t) \) to produce an output \( y(t) \).

- The output, \( y(t) \) will depend on:
  1. the nature of the transfer function (the nature of the nonlinearity)
  2. the amplitude of the input signal \( x(t) \), e.g., increasing the amplitude of the input may cause the output waveform to change shape.

- A linear transfer function with a unit slope and no offset, also called a thrubox, would yield an output that is exactly the same as the input.

Indexing

- The transfer function may be an algebraic function of \( x(t) \).

- Either to save on computation (though less of an issue these days), or to use a waveshaping function that can’t be expressed algebraically (hand-drawn, or data obtained from elsewhere), the transfer function \( w(x) \) may be saved as a vector, or table.

- A waveshaping table will be indexed with the input, that is, each sample of the signal \( x(t) \) is used as an index to the array \( w(x) \). To do this:
  1. scale \( x(t) \), typically between -1 and 1, so that it’s peak-to-peak amplitude equals the length of \( w(x) \).
  2. offset the values of \( x(t) \) so they are positive and begin with one (1) (since we are using Matlab) so we have positive integers as indices to the table.
  3. interpolate the values of \( w(x) \) when the index given by \( x(t) \) is not an integer.
**Linear Interpolation**

- Rather than rounding (or truncating) the values of $x(t)$ so they are integers, it is preferable, and more accurate to **interpolate** the values of the transfer function $w(x)$.
- If $x = 6.5$, we cannot use it to index $w(x)$, because it is not an integer. In linear interpolation, we take the values of $w(x)$ at index 6 and 7, and “construct a line between them.”
- That is, we determine $w(6.5)$ by taking the value that would lie halfway between its neighbouring values, i.e. by scaling each value by one-half, and then adding them together.

**Example of Linear Interpolation**

- If $x = 6.9749$, we may still take the values of $w(x)$ at the surrounding integer indices 6 and 7, but they would be scaled differently, giving greater weight to the 7th element than the 6th.

$$w(6.9749) = (1 - .9749)w(6) + (.9749)w(7)$$

![Linear interpolation](image.png)

**Matlab Linear Interpolation**

- More generally, linear interpolation is given by

$$w(n + \eta) = (1 - \eta)w(n) + (\eta)w(n + 1)$$

where $n$ is the integer part of the original index value, and $\eta$ is the fractional part, indicating how far from $n$ we want to interpolate,

$$\eta = x - n.$$

- Below is a Matlab function which implements linear interpolation.

```matlab
function y = lininterp(w, x);
% LININTERP Linear interpolation.
% Y = LININTERP(W, X) where Y is the output, 
% X is the input indices, not necessarily 
% integers, and W is the transfer function 
% indexed by X.

n = floor(x);
eta = x-n;
w = [w 0];
y = (1-eta).*w(n) + eta.*w(n+1);
y = y(1:length(x));
```

**Thru Box**

- With a thru box, we define a waveshaping transfer function that will do nothing to the signal.
- What is the shape of such a transfer function?
- Though this may not seem very interesting, it’s a good first step in understanding of how we use our waveshaping function and also to make sure we’ve properly implemented linear interpolation.

```matlab
fs = 8;
dur = 1;
nT = [0:1/fs:dur-1/fs];
N = length(nT);
x = cos(2*pi*(1/dur)*nT); % input
xsc = (x + abs(min(x)));
xsc = xsc/max(xsc)*(N-1) + 1; % scale x
w = linspace(-1, 1, N); % waveshaper
y = lininterp(w, xsc);
```
Inverting Box

- Changing the direction of our linear function, we get a waveshaping function that inverts the signal.

```matlab
fs = 1024;
dur = 1;
nT = [0:1/fs:dur-1/fs];
N = length(nT);
x = cos(2*pi*(1/dur)*nT); % input
xsc = (x + abs(min(x))); % offset x
xsc = xsc/max(xsc)*(N-1) + 1; % scale x
w = linspace(1, -1, N); % waveshaper
y = lininterp(w, xsc);
```

Attenuator Box

- We can also make an attenuator by changing the slope of our linear function.

```matlab
fs = 1024;
dur = 1;
nT = [0:1/fs:dur-1/fs];
N = length(nT);
x = cos(2*pi*(1/dur)*nT); % input
xsc = (x + abs(min(x))); % offset x
xsc = xsc/max(xsc)*(N-1) + 1; % scale x
m = 0.5;
w = m*linspace(-1, 11, N); % change slope.
y = lininterp(w, xsc);
```
Transfer Function

- A waveshaper is characterized by its transfer function which relates the input signal to the output signal, that is, the output is a function of the input.
- It is represented graphically with the input on the x-axis and the output on the y-axis.

Even and Odd Transfer Function

- When the transfer function is an odd function the spectrum contains only odd-numbered harmonics.
- When the transfer function is even, the spectrum contains only even-numbered harmonics, thereby doubling the fundamental frequency and raising the pitch of the sound by an octave.
Controlling the spectrum

- A waveshaper with a linear transfer function will not produce distortion, but any deviation from a line will introduce some sort of distortion and change the spectrum of the input.
- To control the maximum harmonic in the spectrum (say, for the purpose of avoiding aliasing), a transfer function is expressed as a polynomial:

\[ F(x) = d_0 + d_1x + d_2x^2 + \ldots + d_Nx^N \]

where the order of the polynomial is \( N \), and \( d_i \) are the polynomial coefficients.
- When driven with a sinusoid, a waveshaper with a transfer function of order \( N \) produces no harmonics above the \( N \)th harmonic.
- The amplitudes of the various harmonics can be calculated using the right side of Pascal’s triangle when the driving sinusoid is of unit amplitude.

Building Pascal’s Triangle

- To fill in the values follow the following two steps:
  1. Set a value in column \( h_0 \) to twice the value of \( h_1 \) from the previous row.
  2. Add two adjacent numbers in the same row and place the sum below the space between them, on the next row.
- Since to start off we only have ‘1’s on the diagonal, begin by filling in a value of 2 for \((x^2, h_0)\).

<table>
<thead>
<tr>
<th>DIV</th>
<th>( h_0 )</th>
<th>( h_1 )</th>
<th>( h_2 )</th>
<th>( h_3 )</th>
<th>( h_4 )</th>
<th>( h_5 )</th>
<th>( h_6 )</th>
<th>( h_7 )</th>
<th>( h_8 )</th>
<th>( h_9 )</th>
<th>( h_{10} )</th>
<th>( h_{11} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.5</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>3</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>8</td>
<td>6</td>
<td>4</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>16</td>
<td>10</td>
<td>6</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>32</td>
<td>20</td>
<td>15</td>
<td>5</td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>32</td>
<td>64</td>
<td>40</td>
<td>25</td>
<td>10</td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>64</td>
<td>128</td>
<td>80</td>
<td>50</td>
<td>20</td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>128</td>
<td>256</td>
<td>160</td>
<td>100</td>
<td>40</td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>256</td>
<td>512</td>
<td>320</td>
<td>200</td>
<td>80</td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>512</td>
<td>1024</td>
<td>640</td>
<td>400</td>
<td>160</td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1024</td>
<td>2048</td>
<td>1280</td>
<td>800</td>
<td>400</td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Finally, to obtain the divider DIV, multiply the value of DIV from the previous row by 2, starting in row \( x^0 \) with 0.5.
Calculating Spectral Output

- Notice from Pascal’s triangle that if the order of the polynomial is even, only even harmonics will be present.
- If the order is odd, only odd harmonics will be present.
- If the transfer function $F(x) = x^5$ is driven by an oscillator of amplitude 1, the output will contain the first, third and fifth harmonics with the following amplitudes:

  \[ h_1 = \frac{1}{16}(10) = 0.625 \]
  \[ h_3 = \frac{1}{16}(5) = 0.3125 \]
  \[ h_5 = \frac{1}{16}(1) = 0.0625 \]

Transfer function $F(x) = x^5$

- Create a 1 second long 220 Hz sinusoid input $x$ and plot the output $y = x^5$ in Matlab:

  ```matlab
  fs = 44100;
  nT = 0:1/fs:1;
  x = sin(2*pi*220*nT);
  y = x.^5;
  ```

  ![Magnitude Spectrum](image)

  Figure 9: Output spectrum of the transfer function $y = x^5$, where $x$ is a unit amplitude sinusoid at a frequency of 220 Hz.

Transfer Functions with Multiple Terms

- If we wish to have a transfer function with multiple terms, then the output will be the sum of the contributions of each term.
- For example, the transfer function

  \[ F(x) = x + x^2 + x^3 + x^4 + x^5 \]

produces an output spectrum with the following harmonic amplitudes:

  \[ h_0 = \frac{1}{2}(2) + \frac{1}{8}(6) = 1.75 \]
  \[ h_1 = 1 + \frac{1}{4}(3) + \frac{1}{16}(10) = 2.375 \]
  \[ h_2 = \frac{1}{2}(1) + \frac{1}{8}(4) = 1.0 \]
  \[ h_3 = \frac{1}{4}(1) + \frac{1}{16}(5) = 0.5625 \]
  \[ h_4 = \frac{1}{8}(1) = 0.125 \]
  \[ h_5 = \frac{1}{16}(1) = 0.0625 \]

  ![Magnitude Spectrum](image)

  Figure 10: Output spectrum of the transfer function $y = x + x^2 + x^3 + x^4 + x^5$, where $x$ is a unit amplitude sinusoid at a frequency of 220 Hz.
Non-sinusoidal input

- The previous calculations are based on a unit-amplitude sinusoidal input.
- Non sinusoidal input to the waveshaping function produces less predictable output, and therefore is more difficult to keep alias free.
- It is, however, possible to change the amplitude of the sinusoidal input so that it is less than—or greater than—1.
- This creates a distortion index similar to the modulation index seen in FM synthesis.

Distortion Index

- If the input cosine has an amplitude of \( a \), then the output in polynomial form becomes
  \[
  F(ax) = d_0 + d_1 ax + d_2 a^2 x^2 + \ldots + d_N a^N x^N
  \]
- Example: Given the waveshaping transfer function
  \[ F(x) = x + x^3 + x^5, \]
  an input sinusoid with amplitude \( a \) yields the output
  \[ F(ax) = ax + (ax)^2 + (ax)^5, \]
  with the amplitude of each harmonic calculated using Pascal’s triangle to obtain
  \[
  h_1(a) = a + \frac{1}{4} 3a^3 + \frac{1}{16} 10a^5 \\
  h_3(a) = \frac{1}{4} a^3 + \frac{1}{16} 5a^5 \\
  h_5(a) = \frac{1}{16} a^5
  \]
- Because an increase in \( a \) (typically having a value between 0 and 1) produces a richer output spectrum, it is often referred to as a distortion index (analogous to the index of modulation in FM synthesis).

Selecting a Transfer Function

- Spectral Matching: Select a transfer function that matches a desired steady-state spectrum for a particular distortion index \( a \).
- This may be done using Chebyshev polynomials of the first kind, denoted \( T_k(x) \), where \( k \) is the order of the polynomial.
- The zeroth- and first-order Chebyshev polynomials are given by
  \[
  T_0(x) = 1 \\
  T_1(x) = x
  \]
  and higher-order polynomials are given by
  \[
  T_{k+1}(x) = 2xT_k(x) - T_{k-1}(x).
  \]
- These polynomials have the property that when a sinusoid of unit amplitude is applied to the input, the output signal contains only the \( k^{th} \) harmonic.

The first few Chebyshev Polynomials of the first kind

- For your convenience, here are some of the first few:
  \[
  T_0(k) = 1 \\
  T_1(k) = x \\
  T_2(k) = 2x^2 - 1 \\
  T_3(k) = 4x^3 - 3x \\
  T_4(k) = 8x^4 - 8x^2 + 1 \\
  T_5(k) = 16x^5 - 20x^3 + 5x
  \]
Matching a Spectrum Using Chebyshev Polynomials

- A spectrum containing several harmonics can be matched by combining the appropriate Chebyshev polynomial for each harmonic into a single transfer function.
- Let \( h_j \) be the amplitude of the \( j^{th} \) harmonic, and \( N \) be the highest harmonic in the spectrum. The transfer function is then given by:
  \[
  F(x) = h_0 T_0(x) + h_1 T_1(x) + h_2 T_2(x) + \cdots + h_N T_N(x).
  \]

Example of Spectral Matching

- Given the following spectrum, what would be the transfer function?

  ![Figure 11: A steady state spectrum.](image)

  The spectrum contains the first, second, fourth, and fifth harmonics, with amplitudes 5, 1, 4, 3, respectively.

Selecting a Polynomial to Fit Data

- The transfer function is given by
  \[
  F(x) = 5T_1(x) + T_2(x) + 4T_3(x) + 3T_5(x) \\
  = 5x + (2x^2 - 1) + 4(8x^4 - 8x^2 + 1) \\
  + 3(16x^5 - 20x^3 + 5x) \\
  = 48x^5 + 32x^4 - 60x^3 - 30x^2 + 20x + 3.
  \]

- If you wish to construct a waveshaper based on incoming data, then you will create a table, and proceed using linear interpolation (as shown in previous slides).
- The problem with this approach is that you can’t ensure a bandlimited spectrum without aliasing.
- It is also possible to fit a polynomial to the data (there are many ways of doing this, the details go beyond the scope of this class).
- You may like to take advantage of Matlab’s \texttt{polyfit} to accomplish this task.