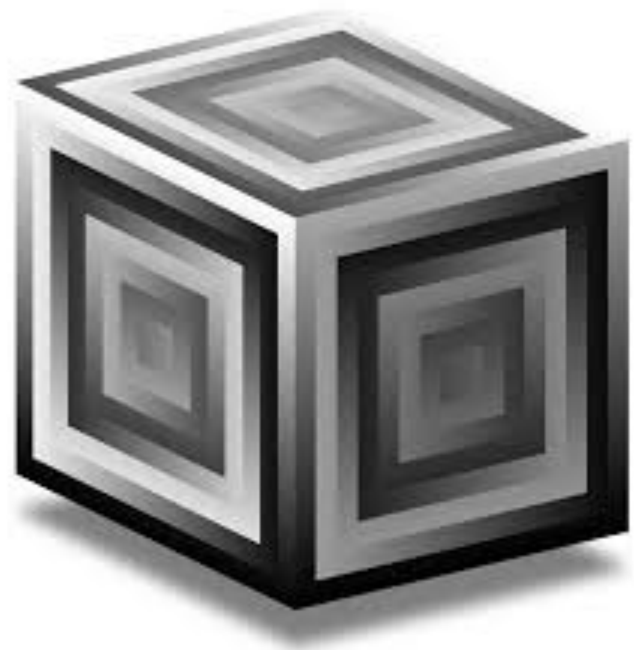
What is Sound?

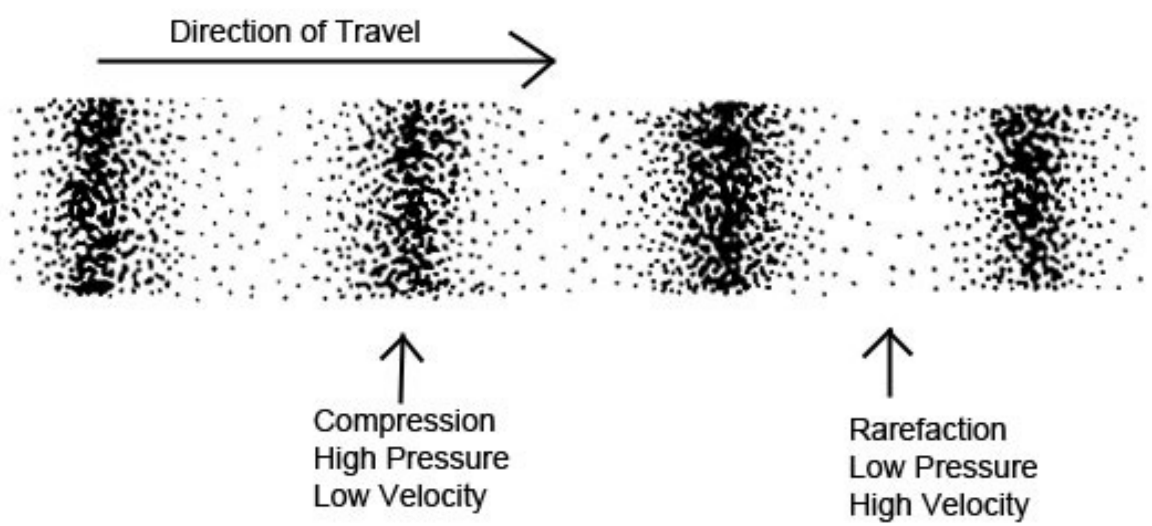
Topics Addressed

- Basic Physics of Sound
- Unit Generators
- Harmonic Series
- Amplitude/Loudness
- Phase
- Balanced/unbalanced cables



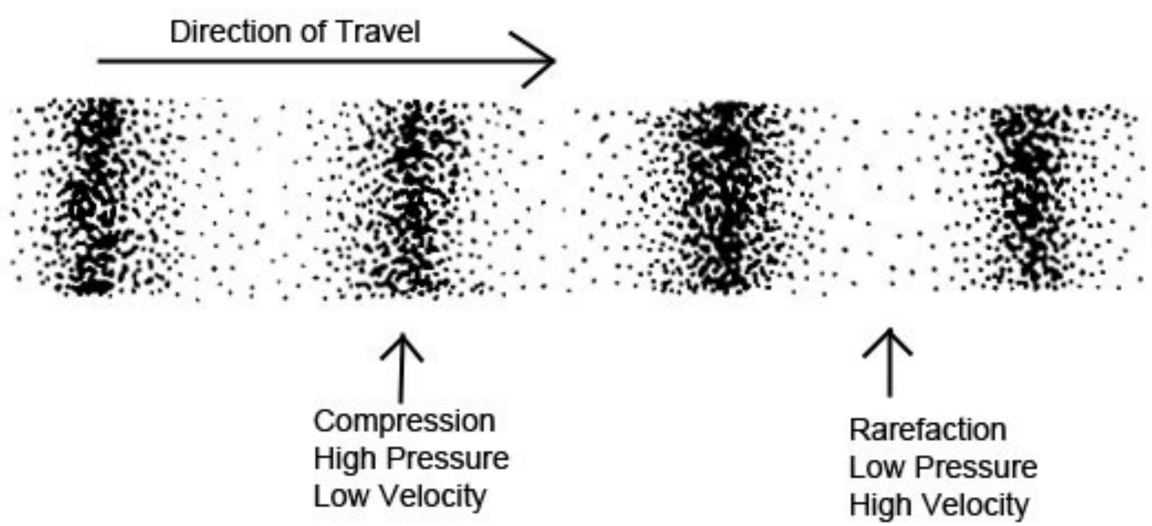
Basic Physics of Sound

- The brain interprets air pressure as sound
- Sound often comes in “waves” of high pressure and low pressure



What is Pitch?

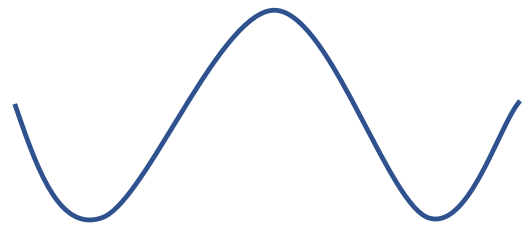
- Pitch is a subjective sensory experience of a physical phenomenon. It is a perceptual property of sound.
- Pitch is bands of compressed air moving at the same regular interval. It is the **frequency** of the sound.
- Higher pitches have a higher frequency. Lower pitches have a lower frequency.

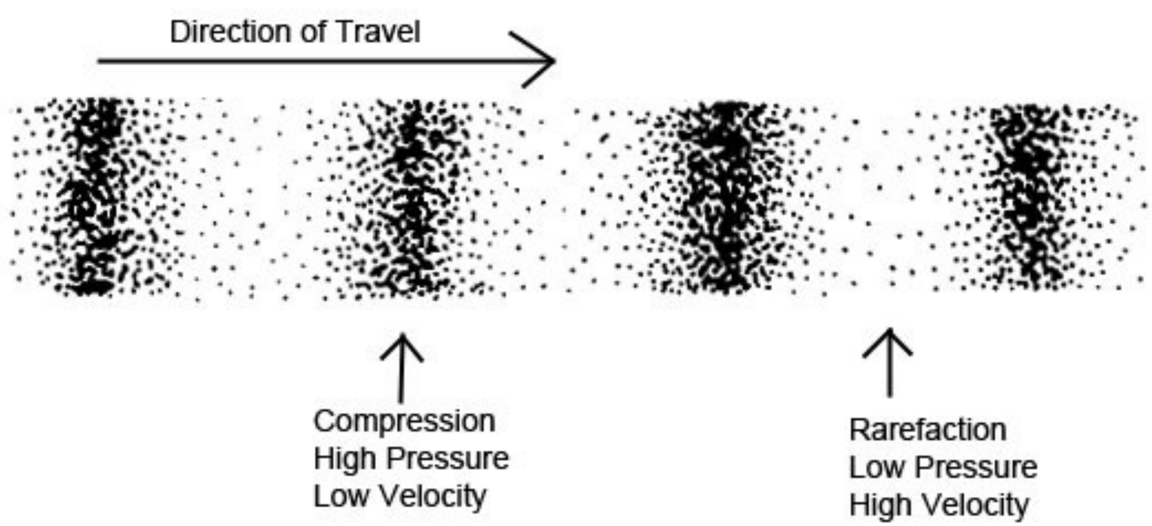


Representing Pitch Mathematically

Sine Wave:

- Has frequency
- Periodic





Create a Sine Wave in SuperCollider

- SuperCollider has a class called `SinOsc` which will produce a sine wave at a specified frequency.
- If that object is wrapped in a function and applied the `.play` method, then an OSC message will be sent to scsynth to produce the sound.

Frequency Amplitude

```
{ SinOsc.ar(440, 0, 0.5) }.play;
```

Wrap in function

Produce at audio rate

Phase

Send msg to server to play

Aside: Create a stereo Sine Wave

- If you played the previous example, you may notice that the sound plays on only one speaker.
- By default, sounds are only sent to one speaker. To send the same sine wave to both speakers, pass an array of frequencies.

```
{ SinOsc.ar([440, 440], 0, 0.5) }.play;
```

Note: a quick way to stop the sound in SuperCollider is COMMAND + PERIOD (macOS) or CTRL + PERIOD (PC)

Liminal Frequencies

- The brain will perceive any regular occurring interval as pitch provided it is in the human audible range from 20-20,000 Hz.
- Let's explore with a pulse wave. We will talk more about pulse waves later but for now you can think of it as a blip of sound that occurs at a regular interval (i.e., at a constant frequency).
- What does it sound like with a frequency of 1 Hz?

```
{ Pulse.ar([1, 1], mul: 0.2) }.play;
```

Frequency

Amplitude

Try various frequencies

- Remember the low end of the human audible range is 20hz

```
{ Pulse.ar(4, mul: 0.2) }.play;
{ Pulse.ar(16, mul: 0.2) }.play;
{ Pulse.ar(20, mul: 0.2) }.play;
{ Pulse.ar(30, mul: 0.2) }.play;
{ Pulse.ar(40, mul: 0.2) }.play;
```

Concert A → `{ Pulse.ar(440, mul: 0.2) }.play;`

Sweep through audible range

- To sweep through the audible range we can modulate the frequency of the pulse wave by using an exponential curve.
- The class `XLine` produces an exponential curve. Here it is being passed into frequency argument of the pulse wave.
- Check documentation for other arguments to the pulse wave.

Start frequency Duration

```
{ Pulse.ar(XLine.kr(10, 20000, 6), 0.1, 0.2) }.play;
```

End frequency

Unit Generators

- What are `Pulse` and `SinOsc`?
  - Unit Generators
- Unit Generators are the sclang representation of a sound
  - A unit generator stores the necessary data for a particular signal but does not actually produce the sound
  - A unit generator's data is sent via OSC messages to the server (scsynth) to actually produce the sound
- Unit Generators come in one of three forms
  - `.ar` -> audio rate
  - `.kr` -> control rate
  - `.ir` -> initialization rate
- We often abbreviate Unit Generators as UGens.

Audio Rate vs. Control Rate

- We won't concern ourselves with initialization rate, but it is important to understand the distinction between audio rate and control rate.
- Specifying a UGen as audio rate implies that the UGen is intended to produce sound.
  - Ex: `{SinOsc.ar(440)}.play;`
- Specifying a UGen as control rate implies that the UGen is intended to modulate an argument to another sound.
  - Ex. `{ Pulse.ar(XLine.kr(10, 20000, 6), 0.1, 0.2) }.play;`
  - Here the `XLine` controls the frequency for the Pulse wave that produces the sound.
  - We could have used `.ar` for the `XLine` but that is inefficient
- Control rate UGens use fewer resources and are less computationally expensive than their audio rate counterparts.

More on pitch

- A simple frequency – represented mathematically by a sine wave – is the basic building block of sound.
- When a frequency lies between 20-20000Hz, our brains will perceive that frequency as a pitch.
- But how then do we explain then the distinction between an oboe playing a concert A and a flute playing a concert A?
  - We described this quality (i.e., how a note sounds) as timbre.
  - There must be some physical difference between the two sounds. What is it?

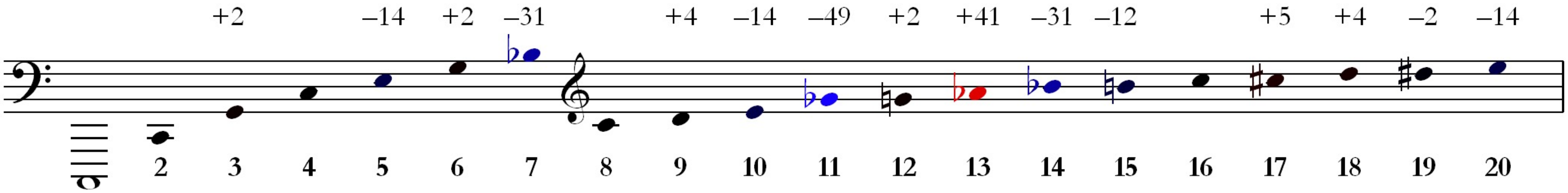
A single pitch is often many pitches!

- It turns out that when we hear a note that it is often comprised of many sine waves that are perceived by the brain as a single note.
- The mixture of these different sine waves creates the perception of timbre.
- In almost all cases, the fundamental frequency (i.e., the note we are perceiving) is one of the various sine waves. Note: it is possible to perceive a given pitch without its fundamental.
- We call these additional sine waves partials.
- What are the other sine waves? What are their frequencies? What are their amplitudes?

Harmonic Series

- Most pitches that we perceive are complex sounds. A complex sound is simply a note that consists of two or more sine waves.
- We call the individual waves in a complex sound partials.
- When we perceive a pitch in a complex sound there tend to be certain partials that are emphasized. We call this pattern the harmonic series and each partial in this pattern a harmonic or harmonic partial.
  - A harmonic has a specific mathematical definition relative to the fundamental frequency. If the fundamental frequency has frequency *x*, then the nth harmonic has frequency *nx*. In other words, a harmonic is a multiple integer of the given fundamental frequency. The harmonic series then is all the frequencies that satisfy *nx* where *n* >= 1 and *n* is an integer.
  - For example: given a fundamental frequency of 100Hz (i.e., the 1st harmonic), then the second harmonic is 200Hz; the third is 300Hz; the fourth is 400Hz… etc.
  - The term overtones refers to all the harmonic partials excluding the fundamental frequency.
- Waves that are part of a complex sound but are not part of the harmonic series are called inharmonic partials.
- In general, a complex sound is perceived by us as having more pitch if there are more harmonic partials than inharmonic partials. More inharmonic partials create a “noisier” sound.

Harmonic Series



EXERCISE: Write a function to print the first 20 harmonics given a fundamental frequency:

```
~produceParitals = {
  arg fundFreq = 40, numHarmonics = 20;
  var harmNum = 1, freq = fundFreq;

  while({harmNum <= numHarmonics}, {
    freq.postln;
    freq = freq + fundFreq;
    harmNum = harmNum + 1;
  });
}
```

Harmonic Series

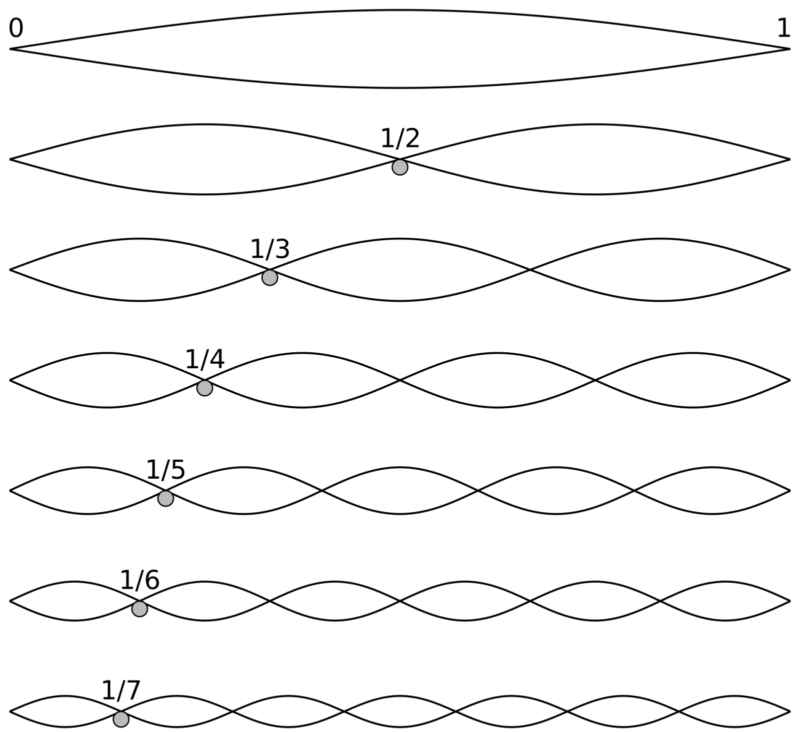
```
r = Routine({
  var harmNum = 1;
  var freq = fundFreq;

  while({harmNum < numHarms}, {
    var sig;
    ("Freq is " ++ freq ++ " | Harmonic is " ++ harmNum).postln;
    sig = {SinOsc.ar([freq, freq], 0, 0.3)}.play(fadeTime: 0.2);
    harmNum = harmNum + 1;
    freq = freq + fundFreq;
    0.7.wait;
    sig.release();
    0.2.wait;
  });
}).play;
```

Play the harmonic series!

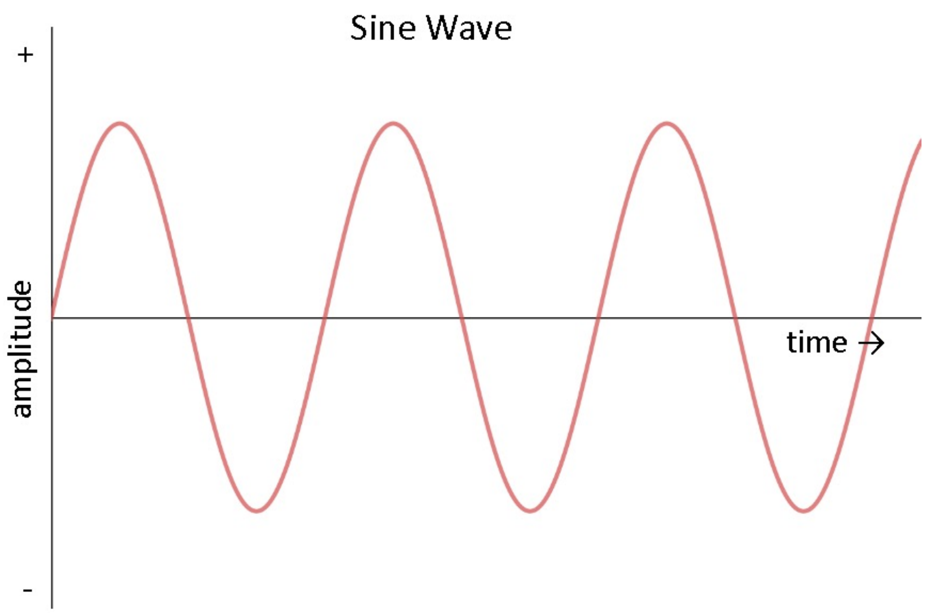
NOTE: This code contains syntax from sclang that we will learn in the coming weeks!

Harmonic Series

- The harmonic series occurs naturally in many types of instruments including wind and string instruments.
- The nth harmonic partial can be found by dividing the string into n parts
- String instruments can produce a pure harmonic by lightly touching the fretting finger at any of these nodes.



Amplitude of Partials



- The amplitude of a soundwave ranges from -1.0 to 1.0.
- Amplitude corresponds to our perception of volume/loudness
- Does the amplitude of harmonic partials affect how we perceive pitches? Does it change the pitch? Does it change the timbre?

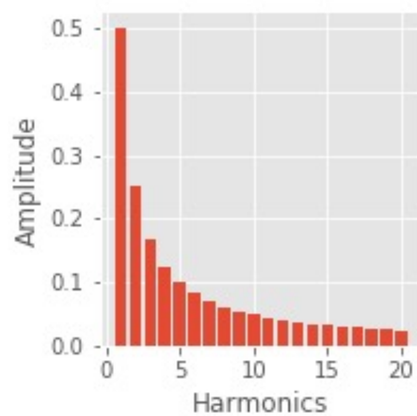
Experiment #1

- What would it sound like if we play a fundamental note (say concert A which is 440Hz) and the first 29 of its overtones with the same amplitude?

```
(
var fundFreq = 300;

for(1, 20, {
  arg n;
  var harmFreq = fundFreq * n;
  {SinOsc.ar(harmFreq ! 2, 0, 0.04)}.play;
});
)
```

Experiment #2

- What would it sound like if we play the overtones with progressively smaller amplitudes? For example what if we said for each nth harmonic, its amplitude would be 1/n of the fundamental frequencies amplitude?



Assume a fundamental frequency amplitude of 0.5

Experiment #2

```
(
var fundFreq = 300;

for(1, 30, {
  arg n;
  var harmFreq = fundFreq * n;
  {SinOsc.ar(harmFreq ! 2, 0, 0.08/n)}.play;
});
)
```

This actually has a special name! It's called a sawtooth wave assuming we have the right phases. More soon.

Experiment #3

```
var fundFreq = 300;
var harmonicNumA = 1;
var harmonicNumB = 12; // Try with a higher number
var largeAmp = 0.2; // Try with a lower number
var smallAmp = 0.05; // Try with a smaller number like 0.005

for(1, 30, {
  arg n;
  var harmFreq = fundFreq * n;
  var amp;

  if((n == harmonicNumA) || (n == harmonicNumB),
    {amp = largeAmp},
    {amp = smallAmp}
  );

  {SinOsc.ar(harmFreq ! 2, 0, amp)}.play;
});
```

What if we just have two partials (harmonics 1 and 12 arbitrarily) that have strong amplitudes and the rest are small?

Experiment #4

What if we remove the fundamental? Does the pitch change? If so, how? Compare this against Experiment #2. This is a sawtooth wave without the fundamental.

```
(
var fundFreq = 300;

for(2, 30, { // Note how we start with the second harmonic
  arg n;
  var harmFreq = fundFreq * n;
  {SinOsc.ar(harmFreq ! 2, 0, 0.08/n)}.play;
});
)
```

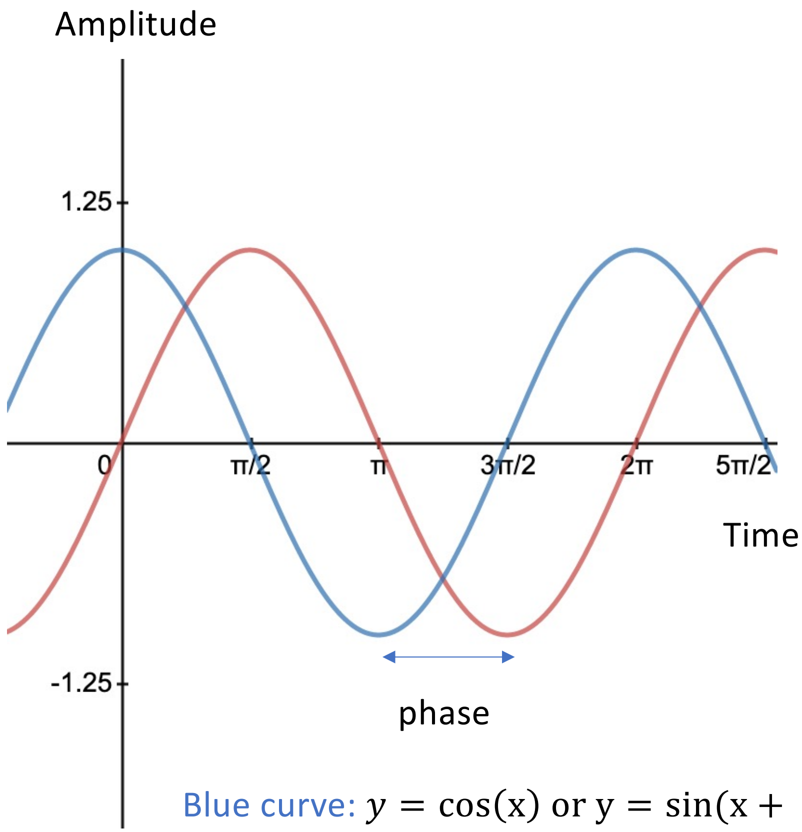
http://auditoryneuroscience.com/pitch/missing-fundamentals

Conclusions

- The ear tries really hard to fuse the harmonic partials into one sound.
- It takes drastic changes in amplitudes to disrupt this perception.
- Remember this is a perception! There really are multiple sine waves occurring at the same time.
- What happens when you introduce inharmonic tones? Explore on your own!

Phase

- Sine waves contain three distinct components: frequency, amplitude, and phase.
- Phase refers to how far the wave is "shifted"
- In mathematics, a sine wave shifted by $-\pi/2$ radians or by $3\pi/2$ radians results in a cosine wave.
- We would say the two waves are 90° or $\pi/2$ radians out of phase.



Blue curve: $y = \cos(x)$ or $y = \sin(x + \frac{\pi}{2})$

Red curve: $y = \sin(x)$

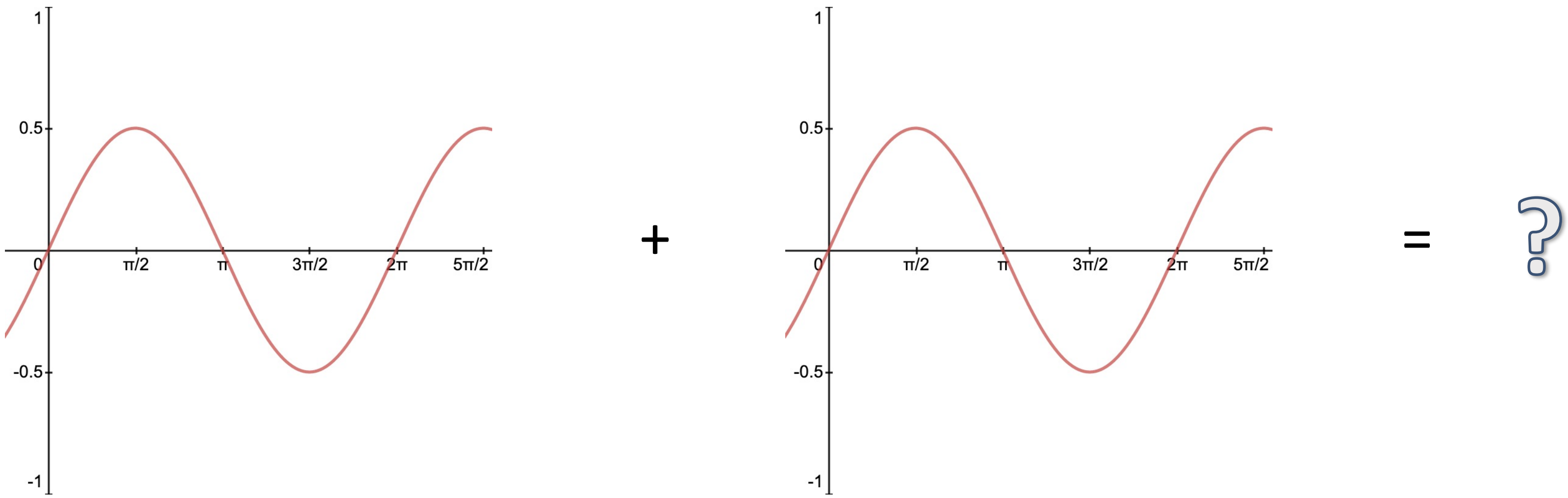
Phase of a Single Wave

- Shifting a single sine wave does not result in a change to our perception.
  - It would still be the same frequency(s) and the same amplitude(s).
- Let’s test it! We should not hear any difference.

```
// A normal sine wave
{SinOsc.ar([440, 440], 0, 0.6)}.play;

// A sine wave shifted back to make a cosine wave
{SinOsc.ar([440, 440], -pi/2, 0.6)}.play;
```

Two Waves in Phase

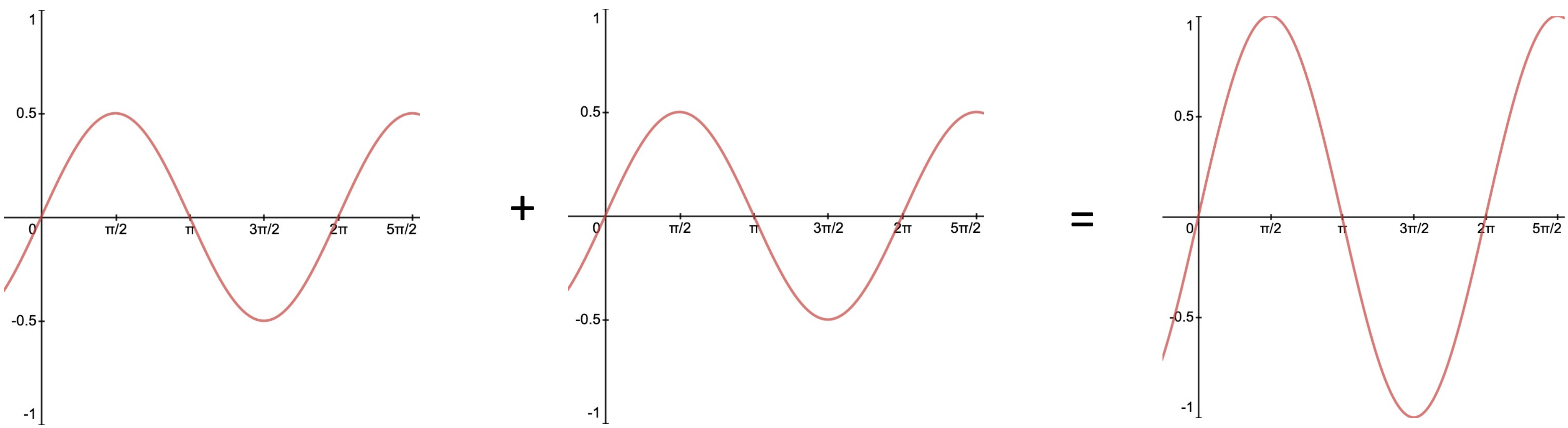
- Phasing becomes interesting when we consider two or more waves.
- Let's take two sine waves each with amplitude of 0.5.
- What do we think will happen?



Two Waves in Phase

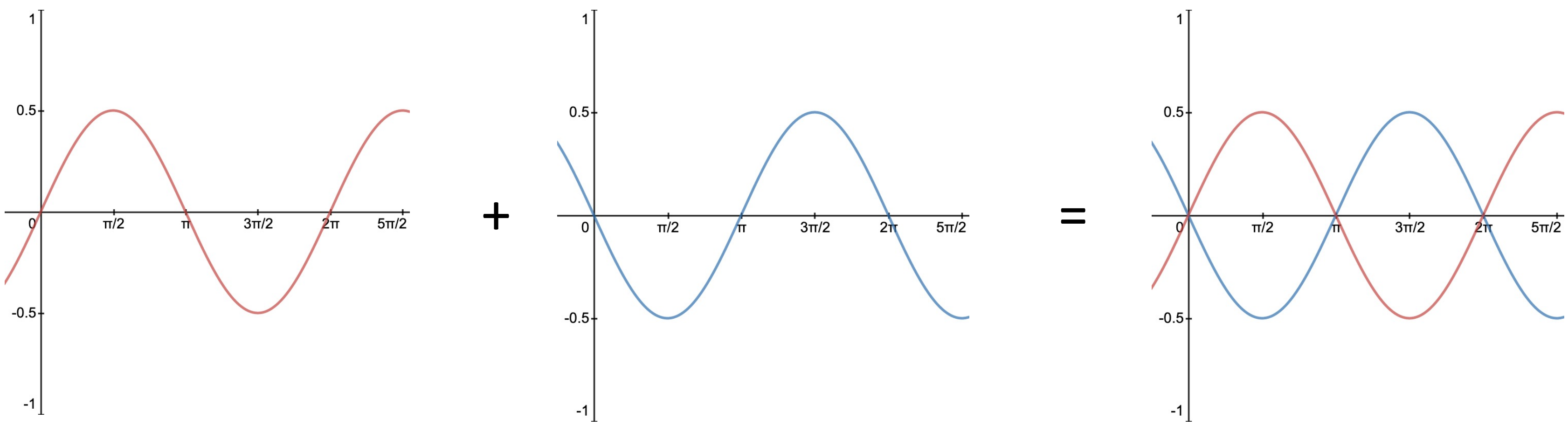
- Two sine waves in phase will double in amplitude
- Waves “in phase” will become louder because their amplitude increases

```
{SinOsc.ar([440, 440], 0, 0.5) + SinOsc.ar([440, 440], 0, 0.5)}.play;
```



Two Waves Out of Phase

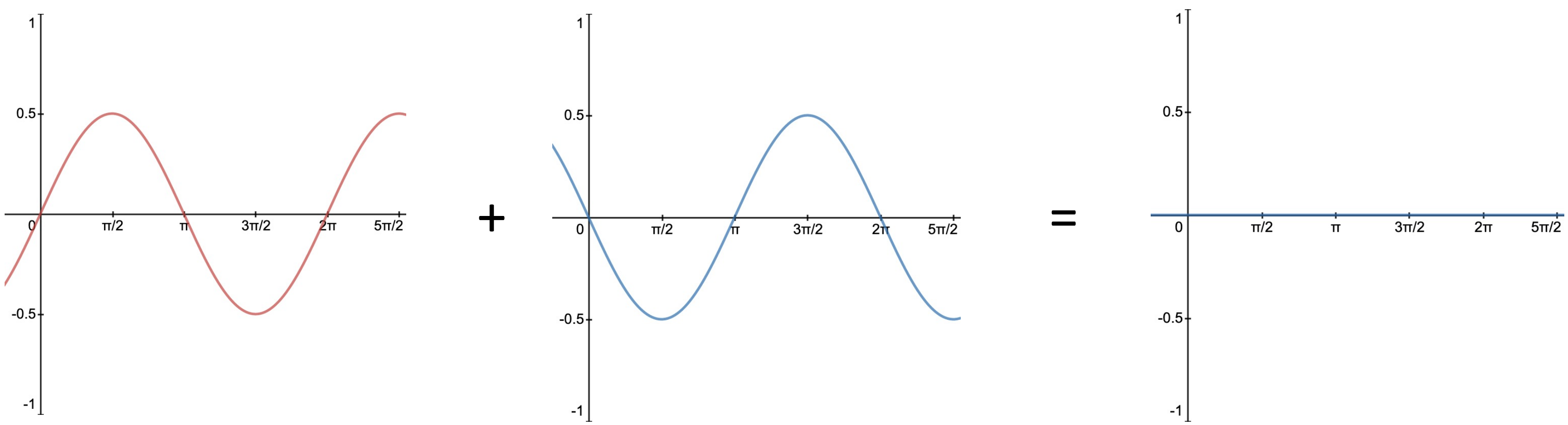
- We saw that adding two sine waves in phase doubled the amplitude of the wave. It did not change the phase of either wave or their frequency.
- What happens if we try to play two sine waves out of phase? Remember this is equivalent to adding the two waves together.



Two Waves Out of Phase

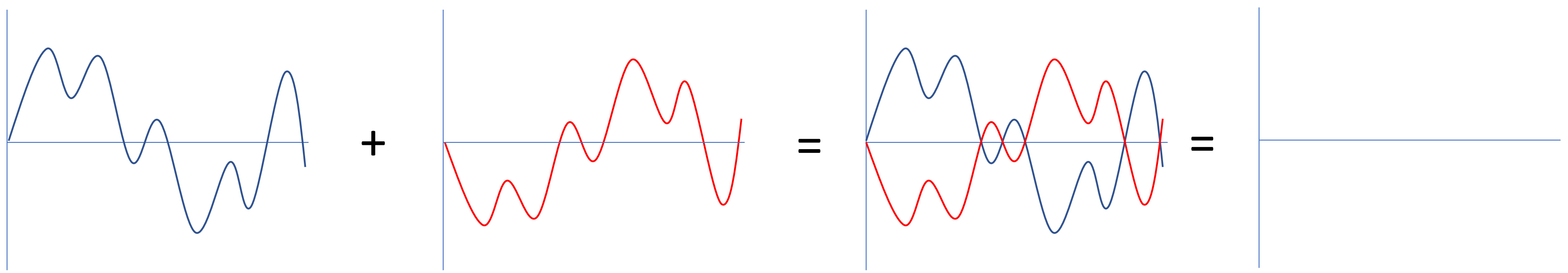
- When two waves are completely out of phase, we hear no sound!
- Summing the amplitudes vertically amounts to summing to zero at any given point in time (the *x* axis)

```
{SinOsc.ar([440, 440], 0, 0.5) + SinOsc.ar([440, 440], -pi, 0.5)}.play;
```

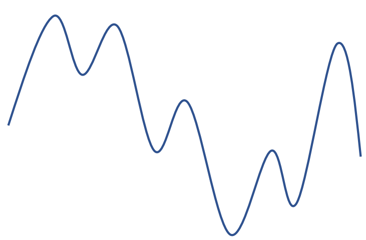


Aside: Balanced Cables

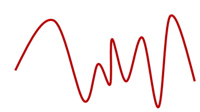
- Problem: many musical cables are susceptible to noise as the signal passes through the wire, distorting the original waveform.
- Solution: balanced cables which take advantage of signal cancelling. Here though, waves are not shifted but rather inverted.
- Inverting a waveform changes its polarity. You can think about this as reflecting the waveform across the x-axis.



Aside: Unbalanced Cables



+



=

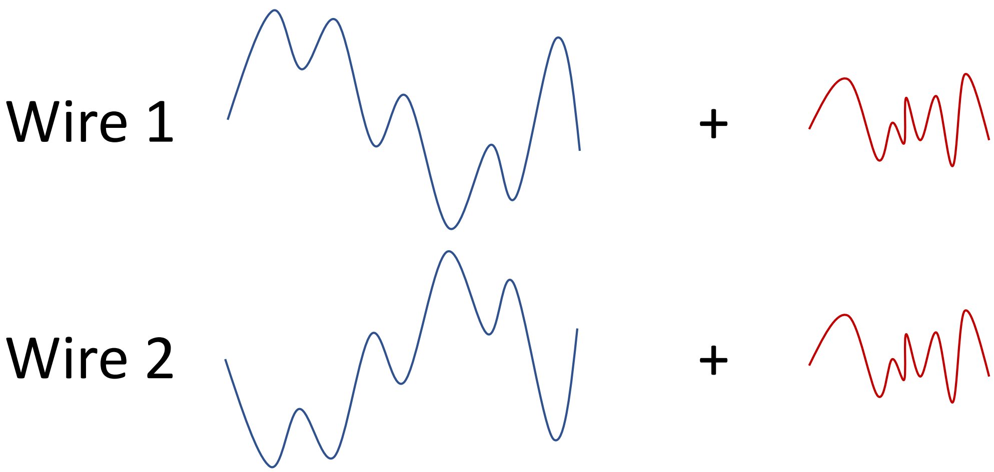
Distorted Signal

Original Signal

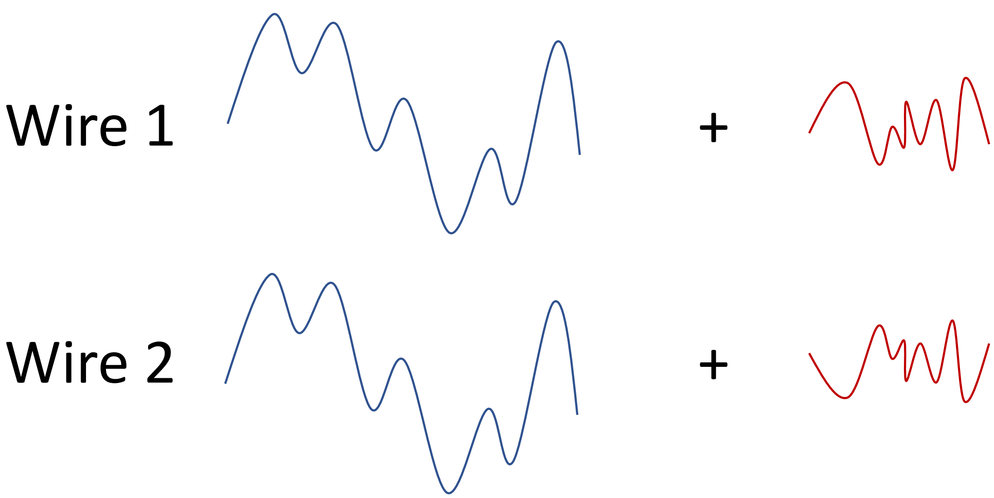
Noise

Aside: Balanced Cables

- Send two signals along two wires but switch the polarity of one of the signals.
- When the signal arrives flip the polarity of the signal that was first inverted.
- This will cancel out the noise and leave the original signal intact.



Wire 2 has its polarity inverted initially. As the signal traverses through both wires, each wire picks up the same noise signal, distorting both signals.



When both wires reach the destination, wire 2's signal is inverted and combine with wire 1. What happens? Well the original signal is in phase and the original noise signal and the inverted noise signal cancel each other out, leaving only the original signal.

Intervals

- Our perception of intervals (octaves, fifths, fourths… etc.) are not based on absolute differences between frequencies but instead on ratios between frequencies.
- Given some frequency f, the octave above f is simply $2f$. This means that a pitch of 200Hz has an octave above at 400Hz and a frequency of 330Hz has an octave above of 660Hz. Note that the absolute differences of 400Hz – 200Hz = 200Hz and 660Hz – 330Hz = 330Hz are **not** the same. We can't simply add, say 200Hz, to every pitch and get an octave.
- Remember this is a perception. Fittingly, note that the first overtone is an octave above the fundamental. The octave is natural to our way of perception and is found in the music of virtually every culture across the globe.

Intervals

- Given that we know that every successive octave is twice the frequency of the previous octave, can we derive a formula to express the frequency of the nth octave given some starting frequency?

$$f = f_{init} * 2^n$$

```
(
var freq = 440, n = 3;
~octaveFreq = {arg freqInit, n; freqInit * (2 ** n)};
{
    SinOsc.ar([freq, freq]) +
    SinOsc.ar([~octaveFreq.value(freq, n), ~octaveFreq.value(freq, n)])
}.play;
)
```

Intervals

- Every interval is simply a set ratio of two frequencies. That set ratio for octaves is two.
- Since intervals are based on ratios of frequencies, our perception is based upon an exponential/logarithmic scale as the formula on the previous slide indicates for octaves.
- There are many systems to determine the precise ratios of intervals. These systems are called temperaments.
  - Equal temperament, the system most commonly used in Western music, sets the frequencies for every note by ensuring that the frequency between any two adjacent notes (i.e., a semitone) is the same. Equal temperament states that 12 semitones must equate to a perfect octave defined here to be twice the frequency.
- For your problem set, you will derive what that ratio number should be and code up a small example to calculate an interval generator given some initial starting frequency.

In summary

- Sine waves are the basic building blocks of sound.
- The important properties of sine waves are amplitude, frequency and phase.
  - Amplitude corresponds to how loud a sound is
  - Frequency corresponds to pitch
  - Phase applies to two or more waves and can create intentional or unintentional noise cancellation or boosting
- Complex sounds consist of two or more sine waves. Many of the pitches that we hear are complex sounds. Complex sounds are perceived as one pitch when the waves mimic the harmonic series.
- It is important to distinguish between the physical realities of sound and the way we perceive sounds.