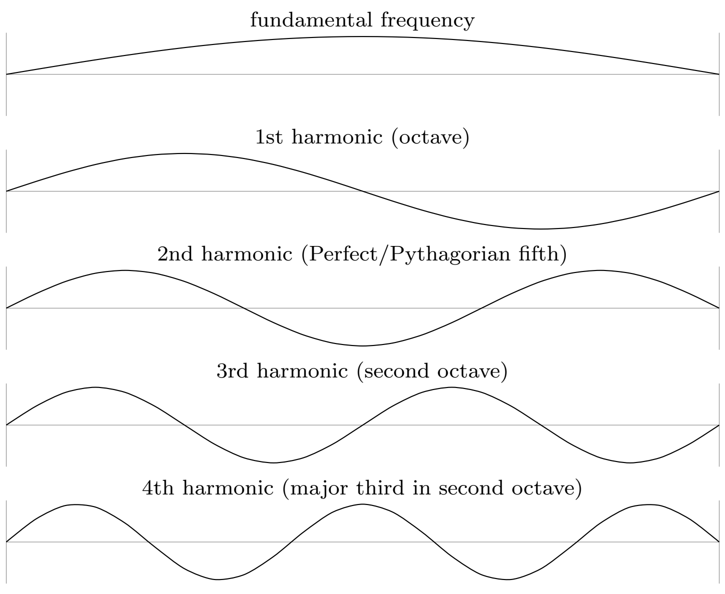
Tuning With Just Intonation and Equal Temparement

Jacob Scott

December 9, 2015

1 A Brief Introduction

The discovery that the freqency of a vibrating string is inversely proportional to its length is generally credited to the Greek polymath Pythagoras in the late 6th century BC [1].
The implication of this discovery was that for any string of length l vibrating at frequency λ hertz, a string of length $\frac{1}{2}l$ will vibrate at 2λ hertz. Doubling the frequency in this way (which can be done by artificially creating a node exactly half way along the string) gives the first harmonic in the harmonic series. The second harmonic would be a string of length $\frac{1}{3}l$ vibrating at 3λ hertz, and so on [4]


fundamental frequency
1st harmonic (octave)
2nd harmonic (Perfect/Pythagorian fifth)
3rd harmonic (second octave)
4th harmonic (major third in second octave)

(for sound, see `https://soundcloud.com/jakescottmusic/harmonics`)

Figure 1: The first four harmonics

(Note: from this point the standard ratio/fraction systems will be used to denote pitches in just intonation. For more information, see [5].)
Pythagoras found that by repeatedly taking the interval between the 1st and 2nd harmonics (i.e. a perfect fifth), all 12 notes of the chromatic scale could be found (for more explanation see the cycle of fifths [2]). However, completing this cycle using 'true' harmonics, the final frequency of the cycle was found to be 'off' from the original, and to compensate, a highly dissonant 'wolf fifth' had to be used to complete the octave. (See `https://www.youtube.com/watch?v=Dl7iIzvUMGg`).
Many different methods of tuning have been invented to try and solve the problem of this apparent error in the fundamental mechanics of music. Broadly, they can be categorised under equal temperament and systems of just intonation.

2 Equal Temperament

The modern system of tuning simply divides the octave into 12 equal steps. A formula for this can be derived as follows:
If P_n is a known frequency and P_a is exactly one octave higher than P_n, then $P_n = 2P_a$.
Since differences in frequency are not additive but multiplicative, $P_n = P_a x^{12}$
Therefore $x^{12} = 2 \implies x = \sqrt[12]{2}$

And from this, $P_n = P_a(\sqrt[12]{2})^{12}$
If the frequency of P_a is known, then P_n can be found using the formula $P_n = P_a(\sqrt[12]{2})^n$. Moreover, if P_n and P_a are n and a semitones from some other 'base' frequency, respectively, then we can use the following formula:

$$P_n = P_a(\sqrt[12]{2})^{n-a}$$

2.1 Cents

The cent is a logarithmic unit in music theory, denoting $\frac{1}{100}$th of a semitone. The advantage of using cents is that they can be used additively to find the difference between two frequencies, which gives a much clearer and more intuitive representation of small intervals.
If we take the previous formula, with $a = 0$, we can derive the formula for the cent.

$$P_n = P_a(\sqrt[12]{2})^n \implies \frac{P_n}{P_0} = 2^{\frac{n}{12}} \implies log_2(\frac{P_n}{P_0}) = \frac{n}{12}log_2 2 \implies n = 12log_2(\frac{P_n}{P_0})$$

Multiplying by 100 to give cents we obtain:

$$c = 1200log_2(\frac{P_n}{P_0})$$

3 Just Intonation

Just intonation is any system of tuning where intervals are given by ratios of 'small' positive integers. Many systems of just intonation have been devised that fit this definition, but for the purposes of this project I will be using five-limit just intonation, and particularly assymetric five-limit just intonation. For more information on types of just intonation, see [5] and [3].

3.1 Five-limit Just Intonation

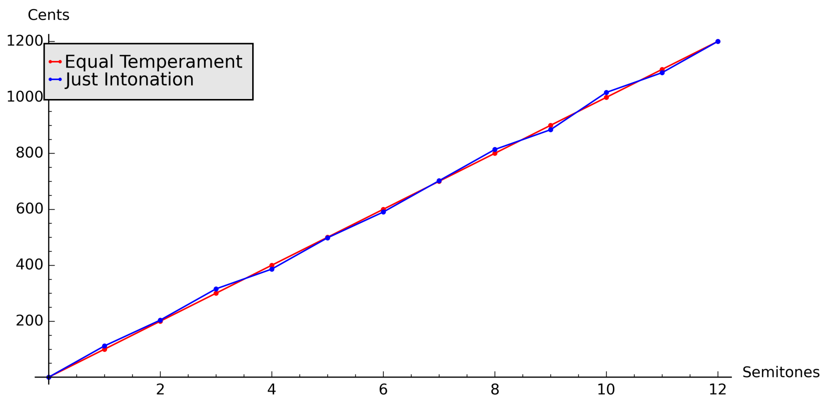
(Note: the following can be found in greater detail online, see [3])

The fractions in Table 1 are found as follows:
For any given note, the corresponding factors are multiplied together to give the the correct fraction for that note at some octave. To obtain the note in its correct octave, the fraction is multiplied by 2 to a chosen integer power.
In the case of E♭, $\frac{1}{5} \cdot 3 \cdot 2^1 = \frac{6}{5}$.
In Table 1, notes with enharmonic equivalents (i.e, two notes that whilst occupying the same position on the keyboard, do not have the same exact frequency) have been shaded with matching colours. To obtain five-limit tuning, the shaded frequencies on the left of the table (i.e. the notes corresponding to fractions 10/9, 16/9 and 64/45) are discarded, giving a full 12-tone justly intonated scale. (Note: there are other systems for eliminating enharmonic equivalents, this one in particular gives *asymmetric* five-limit tuning.)

Factor		1/9	1/3	1	3	9
5	Note	D$^-$	A	E	B	F♯$^+$
	Ratio	10/9	5/3	5/4	15/8	45/32
1	Note	B♭	F	C	G	D
	Ratio	16/9	4/3	1	3/2	9/8
1/5	Note	G♭$^-$	D♭$^-$	A♭	E♭	B♭
	Ratio	64/45	16/15	8/5	6/5	9/5

Table 1: Fifteen Pitches

Sagemath can be used to create a graph showing the differences in cents between equal-tempered and justly intonated pitches, as shown in Figure 2. The complete code used to create the graph can be found at `https://cloud.sagemath.com/projects/bd250a4b-64e4-4530-aa25-0d15b6d41b61/files/pitch%20graph.sagews`



(for an aural comparison see `https://soundcloud.com/jakescottmusic/equal-temperament-just-intonation`)

Figure 2: Difference in cents between justly intonated and equal-tempered pitches

4 Tuning in Python

The following code shows how python can be used to calculate equal-tempered and justly intonated frequencies, and the difference between them in cents. The program the code is taken from has the primary purpose of using randomly generated chords to highlight the differences in these tuning systems, though it also offers the ability to tune chords inputted by a user. The chord Cm^7 (made up of notes C, E♭, G and B♭) in the key of C particularly demonstrates the difference between just intonation and equal temperament. (See Figure 2: E♭ and B♭ correspond to semitones 3 and 10 respectively - points at which the blue line moves particularly far from the red.) The full code can be found at `https://cloud.sagemath.com/projects/bd250a4b-64e4-4530-aa25-0d15b6d41b61/files/tuner%20final%20program.sagews`, but note that the python module 'pyo' must be installed in order to generate audio using the program.

```
freqs_equal = []  # creates an empty list for the final equal-tempered frequencies to be added to
for note in notelist:
    letter = note[0]  # separates 'note' element into its octave and the note's name.
    octave = note[1]
    freq = 440 * ((2 ** (1.0/12.0)) ** (letter - 9 - (12 * 4)))
    # calculates note in its lowest octave, relative to the frequence of A4 (440 Hz)
    if octave != 0:
        freq = freq * ((2 ** (1.0/12.0)) ** (octave * 12))  # transposes the note to its correct octave
    freq = int(round(freq, 0))  # rounds up freq and turns it into an int
    freqs_equal.append(freq)  # adds the found frequency to the list
self.freqs_equal = freqs_equal

freqs_just = []
for note in notelist:
    letter = note[0]
    octave = note[1]  # get the note's name and octave
    if letter - key >= 0:
        interval = letter - key  # calculates interval
    else:
        interval = (letter + 12) - key
        octave -= 1  # calculates interval for a different octave, changes octave to compensate

cents = []  #creates an empty list
for e in range(0, len(notelist)):  #do this as many times as there are notes in the chord
    justfreq = freqs_just[e]
    equalfreq = freqs_equal[e]
    difference = 1200 * math.log((float(justfreq) / float(equalfreq)), 2)
    # uses formula to find difference in cents between two frequencies
    if difference < 0:
        difference = 1200 * math.log((float(equalfreq) / float(justfreq)), 2)
        # if the difference is negative, do it the other way round
    difference = int(round(difference, 0))  # rounds up difference and turns it into an int
    cents.append(difference)  # add the difference to the list
self.cents = cents
```

5 Conclusions

In this report I have briefly explained the difference between equal-temperament and just intonation, giving examples of how the differences can be represented both visually and aurally using sagemath and python. This project could be developed further by using a specifically music-orientated programming language such as ChucK to investigate how the tuning capabilities of computers can be used practically in areas such as composition and performance. In addition to this, many non-Western musical cultures do not split the octave into 12 notes as we do. Exploring how the harmonic series relates to tuning systems such as these could potentially offer much more insight into the mathematics of tuning.

References

[1] John H Chalmers. *Divisions of the tetrachord: a prolegomenon to the construction of musical scales.* Frog Peak Music, 1993.

[2] Wikipedia. Circle of fifths — wikipedia, the free encyclopedia, 2015. [Online; accessed 9-December-2015].

[3] Wikipedia. Five-limit tuning — wikipedia, the free encyclopedia, 2015. [Online; accessed 6-December-2015].

[4] Wikipedia. Harmonic series (music) — wikipedia, the free encyclopedia, 2015. [Online; accessed 5-December-2015].

[5] Wikipedia. Just intonation — wikipedia, the free encyclopedia, 2015. [Online; accessed 5-December-2015].