Max Meets Partch:

Patching Generative Just-Intonation Music in Max/MSP

Kevin Swenson

November 2018

Introduction

In his book Genesis of a Music[1] Harry Partch outlines a practical theory for the realization of music in just-intonation settings. While Partch was able to craft effective music for instruments of his own design and construction such as the Diamond Marimba and Quadrangularis Reversum it is not always a feasible example for other composers to follow since the availability of these instruments is low. Some composers have in turn created there own instruments such as the Bay Area based guitarist Matthew Grasso who designed a guitar to play music in 11-limit just-intonation.[2] Still others have relied on performers to develop an ear for microtonal intervals but that is a very time consuming pursuit. For the author such approaches have proven impractical, as it is too expensive to commission the construction of specialized instruments and too demanding to rely on fellow university students to devote themselves to the mastery of microtonality. Instead, the author has settled on an approach which uses the powerful technologies available to composers today, particularly the computer synthesis program Max/MSP. This paper will outline some of the basic techniques and objects in Max which are applicable to just-intonation, briefly discuss three completed pieces using these techniques and then will describe the conception for a work which is currently in process.

Techniques and Objects

Max/MSP is a program capable of synthesizing almost any sound which may be desired by a given composer. Because of its endless potential for composition it is not within the scope of this paper to give an exhaustive list of the objects which may prove useful for patching pieces in just-intonation. Instead the paper will give an overview of some of the objects which the author has used in the pieces to be discussed later.

The literal production of sound is of the highest importance in composing electronic music.

---

1 Partch, Harry. *Genesis of a Music: An Account of a Creative Work, its Roots and its Fulfillments 2nd Edition,* Da Capo Press, New York 1974.

2 https://www.matthewgrasso.com/guitars.php

Max is filled with objects which function as oscillators and are extremely simple to use. For example, the *cycle~* sine wave generator and *saw~* saw wave generator are basic oscillators capable of accurately producing any desired pitch. The leftmost inlet on both objects takes a message which defines the frequency value in hertz (Hz) for the oscillator. Because oscillator objects in Max can take floating point messages with many decimal points and produce them with extreme accuracy they are ideal for generating microtonal pitch material. To produce more timbrally interesting sounds the *simpleFM* subpatcher can be used to easily craft unique frequency modulated (FM) voices with complex spectra.

Since composition requires the use of pitched material the *coll* object is a powerful tool. *Coll* allows the composer to construct lists of numerical values which can be recalled on command and sent to various objects when required. If, for example, a composer wants to produce just-intonation chords, a *coll* object can easily be used to store all of the necessary values for each chord, where each chord is given its own line in the *coll.* These stored lists of frequencies can then be sent to a series of oscillators to produce just-intonation chords. Individual oscillators can only produce a single pitch at once, so when these values are recalled and sent to a series of oscillators an *unpack* object can be used to separate the individual components of the list.

It is one thing to send frequency values to an oscillator, but for the composition of just-intonation music it is critical to consider how the calculations for the necessary frequencies will be performed. One solution is to allow Max to do all of these calculations in real time. To do this the composer can send a fundamental frequency to multiplication or division (* or /) objects in order to derive the frequencies of desired overtones or undertones. Harmonic *n* is calculated by either multiplying or dividing the fundamental frequency by *n*, multiplying for overtones and dividing for undertones.

This method, however, is only practical if the composer is using frequency values which are harmonics of the fundamental. If the composer wants to calculate frequencies for whole number ratios which represent the space between adjacent harmonics simple multiplication and division will not work. Instead the value of the fundamental frequency must first be multiplied by the numerator of the ratio and then divided by the denominator of the ratio. For example, if the 4/3 perfect fourth is desired the fundamental frequency is first multiplied by 4 and then divided by 3. While it is easy enough to perform these operation inside of Max it becomes much more tedious to patch large numbers of such algorithms, particularly when dealing with high numbered ratios. A 3-limit just-intonation scale, for instance, quickly renders high number ratios: 3/2, 9/8, 27/16, 81/32, 243/64 etc. Therefore, in cases where large numbers of ratios need to be calculated it is best to store the desired values in a coll before hand. Doing so will lead to less cluttered patches and save some processing power.

The free scale editing software *Scala*[3] is a useful tool for quickly calculating frequency values for ratios relative to any fundamental. All one needs to do is dictate the desired fundamental frequency in Hz to the program and input the needed ratios. This method proves extremely handy for quickly transposing a tuning to a new fundamental frequency, as redefining the fundamental will cause the program to instantaneously recalculate the ratios relative to the new fundamental. *Scala* also provides a plethora of tools for the analysis of given tunings as well as comparisons between tunings and a free bank of pre-programed tunings for the user's exploration.

Often the most interesting sounds are produced outside of the digital world, and it is therefore necessary to briefly describe methods of analyzing a signal input into Max. To do this the output of an *adc~* object can be sent to the inlet of a *fzero~* object which detects the fundamental frequency of a signal input. The output of *fzero~* can then be sent to a *flonum* object which outputs floating point decimal values. If programmed properly the value displayed by the *flonum* object will be an accurate representation of the fundamental of the signal being analyzed. This is useful for tuning pitches,

3 http://www.huygens-fokker.org/scala/

particularly on analog synthesizers, and for creating frequency followers which can be used to trigger events inside of Max.

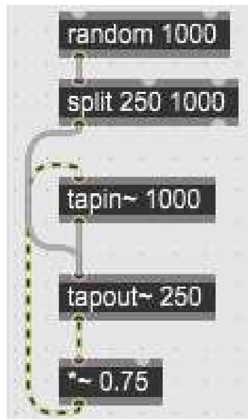
Buchlidian Landscape No. 2

Buchlidian Landscape No. 2[4] is an early drone based composition for Buchla Analog Synthesizer and Max/MSP composed by the author composed in the spring of 2018. The piece is a gradual accumulation of musical material which eventually fades, leaving only the droning fundamental. Individual sounds were layered on top of one another using the *record~* object to store sounds in a *buffer~*. These musical subjects were then looped using *groove~* object to form a dense texture. The piece is in C and all of the pitches are tuned according to the C harmonic series. Two pitches are droning throughout, first a G, tuned as the 3/2 perfect fifth and then the C fundamental. After the drone is established two sequences enter in succession played by the 259e Twisted Waveform Generator which is tuned by the 250e Dual Arbitrary Function Generator. The notes of the sequences expressed as whole number ratios are as follows for both sequences. Sequence 1: 2/1 (C), 6/5 (E-flat), 11/8 (F-sharp), 3/2 (G), 6/5 (E-flat), 17/16 (D-flat). Sequence 2: 3/2 (G), 7/4 (B-flat), 2/1 (C), 17/16 (D-flat), 9/8 (D), 31/30 (D quarter-flat). The 250e sequences were tuned using a cell phone tuner application, and represents a perceivable fault of the piece in relation to the authors more recent work which uses the *fzero~* object inside of Max which is considerably more accurate. Due to this slight inaccuracy exact Hz values for individual notes are not listed for this piece.

There is only a single generative aspect in the piece: a variable delay in Max using the *tapin~* and *tapout~* objects in conjunction with a *random* object. Both of the sequences tuned at the 250e are patched to separate *tapin~* and *tapout~* pairs with a range of 1 second of possible delay. A *random* object with a range of 1000 is patched to a *split* object which is programmed to only let values between 250 and 1000 through its outlet. The random numbers within this range coming from *split* are then sent

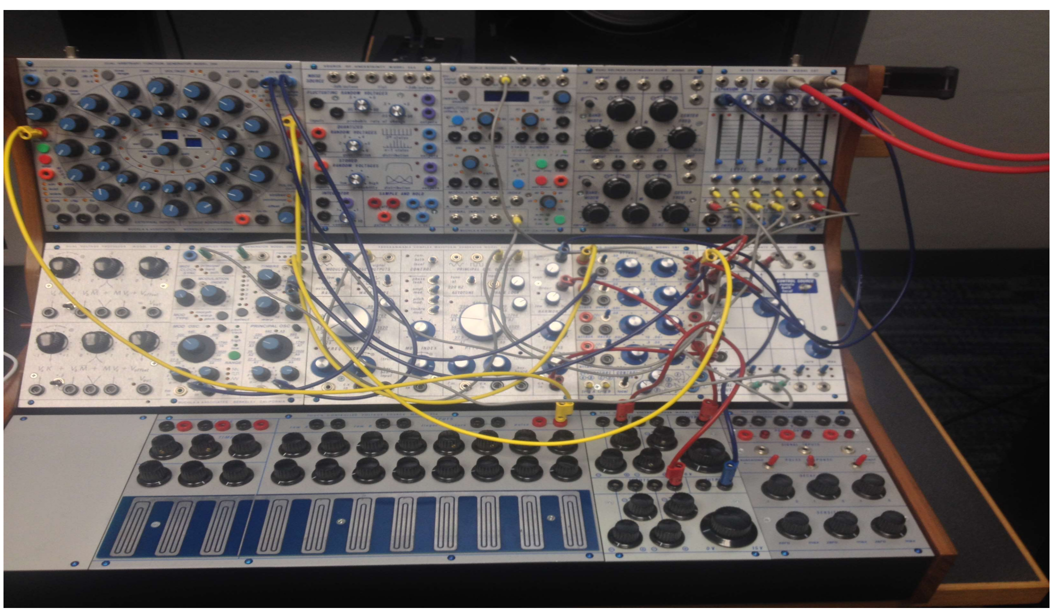
4 http://kevinswenson.com/buchla/buchlidian-landscape-number-2/

to the *tapout~* object producing a random delay for each note in the two sequences. A * object is used to create a feedback loop with the *tapin~* and *tapout~* pairs so that the delay repeats while it gradually decrescendos. Because the delay times are randomized it is impossible for the sequences to sound exactly the same twice. Figure 1 shows the patch used to create this randomized delay effect.



(Figure 1)

Figure 2 shows a picture of the Buchla patch used for the performance of the piece.



(Figure 2)

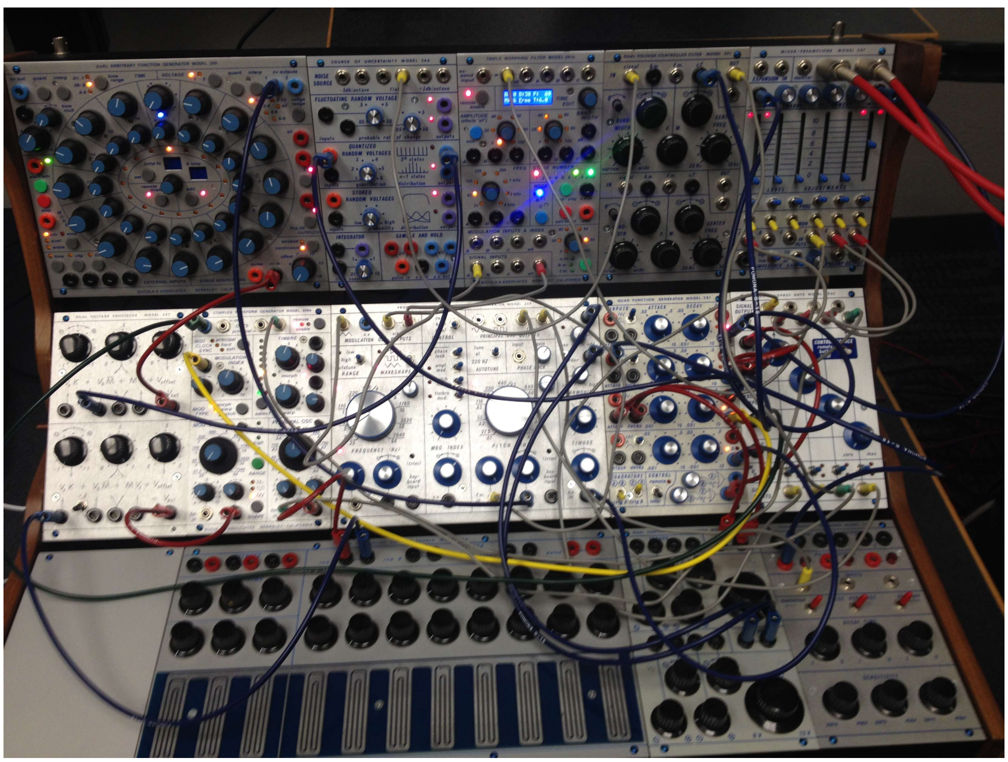
Ritual Dance in a Buchlidian Temple

Ritual Dance in a Buchlidian Temple[5] is another drone based piece for Buchla Analog Synthesizer, which is processed in Max/MSP only with reverb. With a fundamental pitch of D-flat (69.3 Hz), all of the melodic pitches essentially fit into the D-flat Dorian mode. The pitches are tuned according to the harmonics above the 16th, where scales can be constructed purely with the use of natural overtones. Similar to *Buchlidian Landscape No. 2* the piece uses the 250e Arbitrary Function Generator to tune a 16 step sequence played by the 259 Complex Waveform Generator. In addition to the 250e/259 sequence the 216 Voltage Controlled Keyboard is used to tune the 259e Twisted Waveform Generator at the octave for the first 8 notes of the aforementioned sequence. The tuning of the Dorian scale is as follows: 207.9 Hz (3/2 A-flat), 242.55 Hz (7/4 C-flat), 277.2 Hz (1/1 D-flat), 311.85 Hz (9/8 E-flat), 329.175 Hz (19/16 F-flat), 363.825 (21/16 G-flat), 415.8 Hz (3/2 A-flat), 467.7748 Hz (27/16 B-flat), 485.1 Hz (7/4 C-flat), 554.4 Hz (1/1 D-flat), 623.7 Hz (9/8 E-flat), 658.35 Hz (19/16 F-flat), 727.65 Hz (21/16 G-flat), 762.65 Hz (11/8 G), 831.6 Hz (3/2 A-flat), 1108.8 Hz (1/1 D-flat). Unlike *Buchlidian Landscape No. 2* which relied heavily on *record*~ and *groove*~ for its performance, *Ritual Dance in a Buchlidian Temple* does not use Max as an integral component for performing the piece. However, it does use the *fzero*~ object to accomplish much more exact tuning of the desired pitches on the Buchla than in *Buchlidian Landscape No. 2*. While Analog Synthesizers are not really capable of producing the level of exactitude of digital synthesis in terms of intonation, the richness of the timbres they produce makes them a compelling class of instruments for the performance of microtonal music. By combining their sonic capabilities with the analytical power of Max composers can create fairly precise just-intonation compositions much more easily than in the past.

There are generative aspects to *Ritual Dance in a Buchlidian Temple* but they are not generated in Max as they were in the previous piece. Instead the Buchla itself creates elements of randomness

5 http://kevinswenson.com/uncategorized/ritual-dance-in-a-buchlidian-temple/

using the 266 Source of Uncertainty module's Quantized Random Voltage output. This random voltage is patched to the 250e which tunes the 259 Complex Waveform Generator so that each stage of the 16 step sequence is randomly triggered in time with the output of the 266. Another generative element is a product of the 230 Triple Envelope Follower, which follows the envelope of the 259's sequence. The pulse output of the envelope follower is patched to trigger a drum like sound synthesized using a flat noise source from the 266. Carefully adjusting the sensitivity potentiometer for the envelope follower results in only a fraction of the attacks of the 259 triggering a pulse output from the 230 so that an interesting syncopated rhythmic accompaniment with a fair degree of variation is generated. Figure 3 shows a picture of the patch used on the Buchla to perform the piece, in which the elements spoken of can be seen. There is more to the patch than is described here, but it is not entirely relevant to the discussion of either just-intonation or generative musical material.

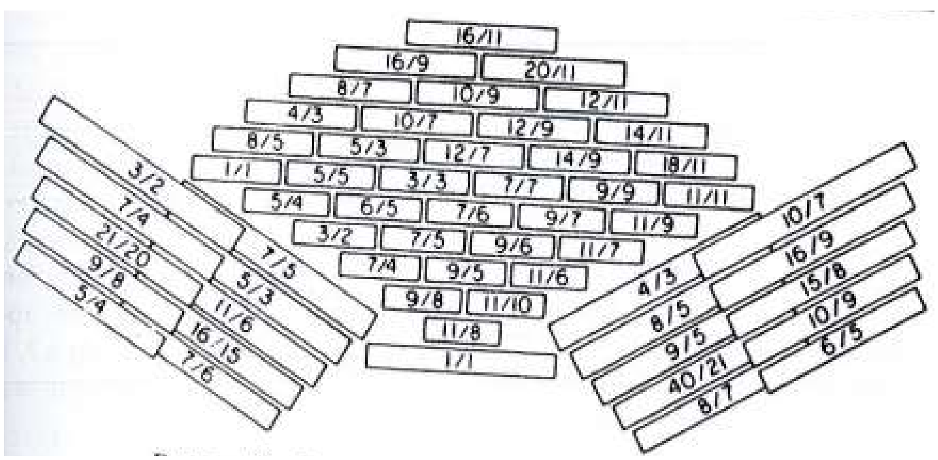


(Figure 3)

Quadrangularis Reversum [6]

Inspired by Harry Partch's marimba like instrument of the same name, *Quadrangularis Reversum* is a piece for solo Viola and Max/MSP. It is in three sections and is composed using a combination of the patching techniques which were used in the two previously discussed pieces. During the 1st section of the piece, the violist records loops of harmonics into *groove~s* over which the they perform a chant like melody in G Lydian. The harmonics used in the loops are the 2/1 octave, the 7/4 minor 7th, and the 9/8 major 2nd. All of the musical material in the 1st section is played by the violist.

In Genesis of a Music, Partch gives the corresponding whole number ratios for each bar on the Quadrangularis relative to the instrument's fundamental pitch. The author used *Scala* to calculate the Hz values for each pitch relative to the C harmonic series. These values were stored in a *coll* object to construct the same chords which are available on the real Quadrangularis. During the 2nd section of the piece, the viola is accompanied by chords derived from the Quadrangularis. Figure 4 shows the diagram of the Quadrangularis from Partch's book which was used to derive the pieces harmonic content.



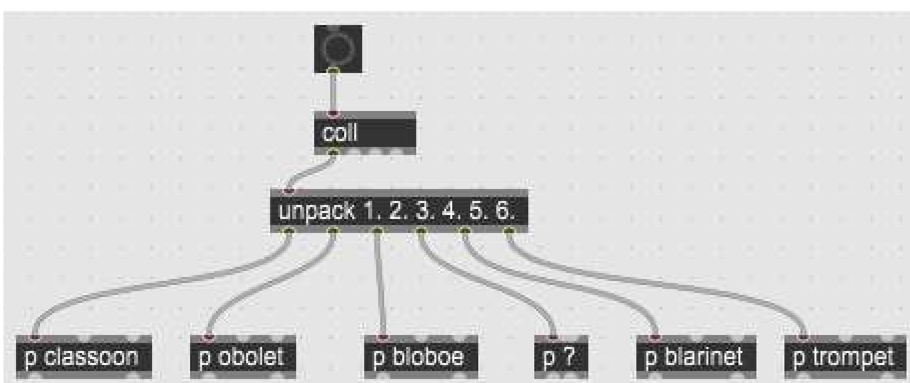
(Figure 4)[7]

6 http://kevinswenson.com/uncategorized/quadrangularis-reversum/

7 Partch, Harry. *Genesis of a Music: An Account of a Creative Work, its Roots and its Fulfillments 2nd Edition,* Da Capo Press, New York 1974. (Page 269)

A frequency follower made with *fzero~* allows the violist to trigger the chords stored in the *coll* by playing from a collection of 16 notated figures. To accomplish this, the individual pitches in each chord are sent to an *unpack* object to separate out each frequency. Then the frequencies are sent to six unique FM voices which serve as a digital pseudo-quadrangularis. Each figure triggers a single FM chord. The order in which the violist chooses to perform the notated figures in the second section is stored by the computer so that the FM chords are performed in the same order during the third section, when the chords are retuned to F. Also during the third section the violist leaves the stage and the Quadrangularis harmonies play on their own while the material that was stored in the groove~'s in the first section returns, retuned to F to fit with the chords.

Figure 5 shows a simplified version of the patch used in *Quadrangularis Reversum* to send Hz values from the *coll* to the FM voices. The *bang* object represents variables going into the inlet of the *coll* object that are triggered by the viola using the frequency follower. The *unpack* object separates the 6 values in each line of the *coll* and sends them to individual FM sub-patchers i.e. classoon, obolet, bloboe etc.



(Figure 5)

The contents of the *coll* object are written as follows from left to right. At the beginning of each line a number representing the symbol which calls up that line, followed by a comma. For example a line that begins "1," will be called if a 1 is sent into the inlet of the *coll*. Following the line number and comma, the individual list elements are written, each one followed by a single space. This way, when a line is sent to *unpack* the spaces tell the program how to separate the values to be sent out of the 6 individual outlets on the *unpack* object. At the end of the line a semi-colon is written in order to tell the program that the line is over. Therefore the contents of the first line of the *coll* used to store the frequency values for the 2nd section of the piece with the Quadrangularis harmonies tuned to a C fundamental are as follows: "1, 1438.94 1177.3144 915.9692 392.4382 654.0636 130.81275;". This line represents the Quadrangularis chord represented by the topmost diagonal moving leftward seen in figure 5 which contains the ratios: 16/11, 16/9, 8/7, 4/3, 8/5, 1/1.

The construction of the frequency follower is critical to the performance of the piece. A microphone is used to send a signal input from the viola to Max using *adc~* which is then analyzed by *fzero~*. The floating point value displayed by the *flonum* object is then patched to a series of if/then statements which send out a specific number to call up the corresponding line stored in the *coll* object. Because it is nearly impossible for a violist to consistently produce exact Hz values for any given note, a range is defined by the if/then statement which compensates for human error. Figure 6 shows the a simplified version of the patch used to trigger line 6 in the *coll*.

(Figure 6)

The if/then statement in figure 4 is triggered by the note B4, which is approximately 493.88 Hz. The expression statement reads that, "if $i1 > 488 && $i1 < 501 then 6 else out2." This translates to: "if integer 1 is greater than 488 Hz and integer 1 is less than 501 Hz then output 6 from outlet 1. If not then send the output from outlet 2." The 6 that is sent out the 1st outlet will call 6 in the *coll* only if the *flonum* value is between the range defined in the if/then statement. If it is outside of the defined range the output will come out of the 2nd outlet, which is not connected to anything in the actual patch and therefore has no effect. This is only one example of an if/then statement used in the frequency follower portion of the patch for *Quadrangularis Reversum*. There are 15 others, each of which send out a specific number from 1-16 triggered by a note within a clearly defined frequency range.

Several elements of the piece are indeterminate both in the viola part and in Max. As mentioned previously, the violist chooses to perform the 16 notated figures which trigger the Quadrangularis harmonies in a semi-random order. The first note which triggers a harmony, B4 always occurs at the same place, but the other 15 triggers are somewhat up to the violist. The remaining 15 figures are arranged in 5 rows of 4 figures, 3 figures, 1 figure, 3 figures and 4 figures respectively. The player must perform the rows of figures in the order which they appear vertically, but they may perform the rows horizontally in any order of their choosing. Figure 7 shows the part of the viola score in which these rows occur.

(Figure 7)

There are several variable components of the Max patch as well. Firstly, during some of the notated figures which the violist plays to trigger a harmony there may in fact be silence even though a line in the *coll* has been recalled. Silence from the Max patch may occur in measure 39, 41, 42, 44, 47, 49 and 51. If no sound occurs in 39 then one will occur in 49 and vice-versa. The same relationship exists between 41 and 51. If no sound occurs in both 42 and 44 then a sound will occur in 47. If a sound occurs in either 42 or 44 then no sound will occur in 47. All of this is accomplished by using *random* objects to send out either a 1 or 0 to open or close a *gate* object in 39, 41, 42 and 44. Depending on whether or not a 1 or 0 was received, if/then statements will either open or close the same *gate* in the manner described above.

The octave of each individual note received by the 6 FM objects is also randomized. *Random* objects call up lines stored in 6 *colls* each time a chord is triggered by the violist. This sends values of 1, 2, 4 or 8 to */* objects which determine the octave of the frequency patched to each FM voice. As a result one can never determine which voice will serve as the bass or soprano of the chords. Since the timbre of the FM sounds changes slightly in each register, subtle changes of color are heard due to the random octave placement.

Generative Meditation

Following the progress made in the 3 previously discussed pieces, the author is currently working on a generative piece which is composed in a just-intonation environment which changes based on the date and time. Using the *date* object in Max/MSP which outputs values for the year, month and calendar day an algorithm was constructed to determine the fundamental frequency of the piece on any given day. The algorithm is: (month/10) + (day/10) + (year/100) = Hz. For January 1st 2018 the fundamental is: (1/10) + (1/10) + (2018/100) = 20.38 Hz. For December 31st 2018 the fundamental is: (12/10) + (31/10) + (2018/100) + 25.48 Hz. Therefore the year 2018 has a range of 5.1 Hz for the fundamental frequency. This range approximately covers E0 to G#0, basically spanning the interval of a major third throughout the year. Towards the end of any given month the frequency will be higher than at the beginning of the month that follows it.

The fundamental frequency is then used to construct a drone using the additive synthesis technique. A series of 16 *cycle*~ objects are used to produce the drone which encompasses the first 16 harmonics of the given fundamental for the day on which the piece is being played. This is accomplished by sending the fundamental frequency value to * objects which produce the frequencies of the desired harmonics i.e. *2 for the 2nd harmonic, *3 for the 3rd harmonic, etc. Each harmonic has an attack-decay amplitude envelope generated by a *line* object that increases from 0 to 100 over 30 seconds and then decays from 100 to 0 over 30 seconds making the total time of the envelope 1 minute for each harmonic. The envelopes are triggered by a bang delayed by 2, 4 or 8 seconds depending on the octave of the harmonic. Bangs are sent to begin the envelope of each consecutive harmonic once the attack portion of the preceding envelope has begun. Once the amplitude envelope for each harmonic of the drone has begun it cycles indefinitely.

Depending on the time of day certain octaves of harmonics may receive an additional delay of the beginnings of their amplitude envelope. This is accomplished by the use of two if/then statements which are patched to the hour output of the *date* object. The first if/then statement reads, "if $i1 > 17 then 0 or $i1 < 8 then 0 else 1." In practical, this means that if the piece is being performed between 5pm and 8am the if/then statement outputs a 0 which closes a gate controlling the delayed bang that would trigger the amplitude envelope for the 5-8 harmonics of the drone. If the gate is closed because the piece is being played during the specified times, then the amplitude envelope of the 5-8 harmonics will only begin after the 1-4 harmonics have completed one full cycle. Similarly, the second if/then statement reads, "if $i1 > 21 then 0 or $i1 < 5 then 0 else 1," which means that if the piece is performed between 10pm and 5am then a gate controlling the delayed bang that would trigger the amplitude envelope for the 8-16 harmonics is closed. This gate is opened once the 4-8 harmonics have completed one full cycle. The net effect of these two extended delays is that it takes longer for the drone to become fully operational as it becomes later into the day or night. Furthermore, the overall shape of the drone can take 3 different forms depending on the time at which the piece is performed. Because the drone is crafted using additive synthesis, the timbral aspects of the drone change depending on the shape of the drone.

The piece is written in a modal environment where the day of the week on which the piece is performed determines the mode which will be used. Monday is Ionian, Tuesday Dorian, Wednesday Phrygian, Thursday Lydian, Friday Mixolydian, Saturday Aeolian, and Sunday Locrian. Each mode is tuned to specific intervals of the harmonic series. Figure 8 shows a chart denoting the harmonics used for the tuning of each mode. Cent values (c) are shown below each ratio in place of Hz, as the variable fundamental frequency makes the use of Hz impractical.

Mode/Day	Do	Ra/Re	Me/Mi	Fa/Fi	Se/Sol	Le/La	Te/Ti
Ionian - (Monday)	1/1 0c	9/8 (Re) 203.9c	5/4 (Mi) 386.3c	4/3 (Fa) 498c	3/2 (Sol) 702c	5/3 (La) 884.4c	15/8 (Ti) 1088.3c
Dorian (Tuesday)	1/1 0c	9/8 (Re) 203.9c	6/5 (Me) 315.6c	4/3 (Fa) 498c	3/2 (Sol) 702c	5/3 (La) 884.4c	7/4 (Te) 968.8c
Phrygian (Wednesday)	1/1 0c	16/15 (Ra) 111.7c	7/6 (Me) 266.9c	13/10 (Fa) 454.2c	3/2 (Sol) 702c	13/8 (Le) 840.5c	7/4 (Te) 968.8c
Lydian (Thursday)	1/1 0c	9/8 (Re) 203.9c	5/4 (Mi) 386.3c	11/8 (Fi) 551.3c	3/2 (Sol) 702c	5/3 (La) 884.4c	15/8 (Ti) 1088.3c
Mixolydian (Friday)	1/1 0c	9/8 (Re) 203.9c	5/4 (Mi) 386.3c	4/3 (Fa) 498c	3/2 (Sol) 702c	5/3 (La) 884.4c	7/4 (Te) 968.8c
Aeolian (Saturday)	1/1 0c	9/8 (Re) 203.9c	6/5 (Me) 315.c	4/3 (Fa) 498c	3/2 (Sol) 702c	13/8 (Le) 840.5c	7/4 (Te) 968.8c
Locrian (Sunday)	1/1 0c	16/15 (Ra) 111.7c	7/6 (Me) 266.9c	13/10 (Fa) 454.2c	11/8 (Se) 551.3c	14/9 (Le) 764.9c	7/4 (Te) 968.8c

(Figure 8)[8] [9]

8 Partch, Harry. *Genesis of a Music: An Account of a Creative Work, its Roots and its Fulfillments 2nd Edition,* Da Capo Press, New York 1974. (Page 461-463 Appendix A)
9 https://www.kylegann.com/Octave.html

These pitches will be performed in Max by an FM synthesis voice whose harmonicity ratio will be in part determined by an algorithm which combines the values for the hour and minute in some way, similar to the way in which the piece's fundamental frequency was derived. The duration of each note played by the FM voice will be quite long and will help to establish the intonation for the day. Only after all the notes of the drone have begun to cycle will the FM voice enter. Then, once the piece begins to settle a solo human instrumentalist will perform a meditative modal improvisation over the electronic material. Currently plans are being made for a flute, violin and alto saxophone player to perform over the patch among others. A given performance will last at least an hour after which some means of ending it will be constructed.

Conclusion

Clearly Max/MSP is a powerful tool for the creation of generative just-intonation music with an extremely high level of accuracy in terms of tuning. Its capabilities are enhanced when used in combination with a human performer, whose ability to articulate phrases with a high degree of passion and musicality far surpasses that of a computer. The techniques shown here are merely a starting point from which further exploration may be undertaken. In addition, the current artistic vision of the author in regards to technology based composition is represented. Using the building blocks established in the 3 completed compositions, the generative meditation will augment their interactive and variable elements. Hopefully by reading about these endeavors and the way they were accomplished, those who are interested in exploring some of the microtonal capabilities of Max have gained some insight as to how they might approach it.

Bibliography

1) Partch, Harry. *Genesis of a Music: An Account of a Creative Work, its Roots and its Fulfillments 2nd Edition*, Da Capo Press, New York 1974.

2) Grasso, Matthew. "Matt's Guitars," *Matthew Grasso - Guitarist* (blog), 2016. https://www.matthewgrasso.com/guitars.php

3) E.F. Op de Coul. "Scala Home Page," *Huygens-Fokker Foundation Centre for Microtonal Music* (blog), October 9, 2018. http://www.huygens-fokker.org/scala/

4) Swenson, Kevin. *The Music of Kevin Swenson* (blog), November 29, 2018. http://kevinswenson.com/

5) Gann, Kyle. "Anatomy of an Octave," *Kyle Gann Composer and Author* (blog), December 15, 2013 https://www.kylegann.com/Octave.html