Intuitive Models for Adaptive Musical Tuning

Stephen Karukas

Abstract. This paper presents two algorithms for the analysis and retuning of musical pitch collections, taking as their models the musical harmonic series and just intonation (JI), respectively. The first finds a portion of the harmonic series that matches the structure of the given pitch collection, retuning each pitch as an integer partial of a fundamental frequency. The second represents each pitch collection as a weighted graph, generating a subset of the graph, a *minimum diameter spanning tree*, that conforms to a certain intervallic hierarchy such as that of 5-limit JI. This approach is shown to produce optimal results even for complex collections including diatonic scales, chromatic scales, and polytonal aggregates. These algorithms are purposefully intuitive, designed to be implemented in a way that exposes their underlying models, allowing for their use as microtonal composition tools.

Introduction: Why adaptive tuning?

Though the phrase *adaptive tuning* may suggest that this paper will present a tool such as Auto-Tune™[1], in a way the methods discussed in this paper actually achieve the opposite: rather than fixing tuning to a prescribed system, they attempt to change the system.

So, if that is adaptive tuning, why? This paper's answer gives an important reason for the types of algorithms introduced herein: *The primary motivation for developing an adaptive tuning system is the potential for new technology to change the way music is created.*

Historically, issues arising in acoustic instrument construction have led to various fixed tuning systems, and digital representation of music unlocks exciting potential for music beyond these systems. This introduction will briefly describe these issues as well as some existing approaches.

Simultaneous musical pitches are generally less dissonant when their frequencies are related by a "pure" ratio composed of small integers, such as 3:2 or 5:4. The most fundamental intervallic unit of Western music is the octave, with a frequency ratio of 2:1, that relates pitches of the same class (i.e. an octave above C is C). However, it is impossible to create a closed system where 5:4 and 3:2 ratios combine in some way to make an octave, as there are no $a, b, c \in \mathbb{Z}^+$ that satisfy the equation $\left(\frac{5}{4}\right)^a \left(\frac{3}{2}\right)^b = 2^c$. Therefore, in order to avoid fitting an infinite number of keys on a piano keyboard (or strings within the body of the piano), ratios other than the octave are stretched or compressed in practice. Modern instruments are generally designed to perform in equal temperament, a system where the octave is divided equally into twelve parts

[1]Auto-Tune™ is a trademark of Antares Audio Technologies: https://www.antarestech.com/product/auto-tune-pro/

(semitones).[2] However, because digital keyboards have no physical strings, tuning can be changed dynamically, making available a continuous palette[3] of pitches rather than a discrete one. An adaptive tuning system is one that allows these continuous pitches to be systematically selected at the time of performance.

Adaptive tuning remains a complex issue, one with no provably perfect solution. The main reason for this is the number of choices one must make when integrating an adaptive tuning system (or any microtonal system, for that matter) into a vast musical ecosystem of instruments, musical works, and theory, all of which is based upon the tried-and-true discrete model of pitch.

Existing Systems

Some existing adaptive tuning systems approach the problem in the most abstract way, such as one devised by William A. Sethares. His algorithm retunes a collection of pitches by finding local minimums of a "sensory dissonance curve" using gradient descent. This is perhaps the most powerful adaptive tuning system, as it can calculate timbre-dependent minimum dissonance even for non-harmonic sounds.[4]

Other systems attempt to adjust tuning as imperceptibly as possible so as to avoid the dangers of audible microtonality, as these systems are designed to be applied to existing tonal music. One example is Hermode tuning, a commercially available adaptive tuning system created by Werner Mohrlok. Its method of analysis is based upon identifying certain chords and intervals and producing "best fit", and occasionally tempered, tunings. [5]

In contrast, the methods presented in this paper do not compromise, or temper, intervals, so all resultant frequencies are related by small integer ratios. The models are also relatively intuitive, producing results whose theoretical bases are transparent ("brute force" as opposed to an approach like Sethares's). Though this is a paper on the topic of algorithms, it should be added that the best implementation of these algorithms would be one in which the underlying model is exposed (and even editable by the user)—current methods for creating music in alternate or dynamic tunings tend to be inflexible, coming with a steep learning curve. Given that the majority of the adaptive tuning process is analysis of a pitch collection, it is imperative that the results of this analysis are made readily available to aid in the process of composition. Otherwise—to return to the introductory question—"why adaptive tuning?".

Matching the harmonic series

Existing research into adaptive tunings has primarily used "*n*-limit" just intonation as its model, where *n* is a small prime number representing the highest partial index

[2] Frequency ratios combine by multiplication, so a semitone has the frequency ratio $\sqrt[12]{2}:1$. When musical pitch (logarithmically related to frequency) is measured in semitones, these values combine by addition and are referred to by their letter names or by integers (C=0, C#=1, D=2, …, B=11). In this paper musical intervals will be identified both by frequency ratios and by the number of semitones their equal-tempered counterpart occupies in "pitch space", as the latter is more common in music theory.

[3] Where the discrete model of pitch uses semitones, continuous intervals are referred to as "microtones", units of no specific size, usually smaller than a semitone.

[4] Sethares, William A. *Tuning, Timbre, Spectrum, Scale*. 2nd ed., London, Springer-Verlag, 2005, pp. 162-178.

[5] Mohrlok, Walter. "The Hermode Tuning System." Self-published, 2003. http://extras.springer.com/2005/978-1-85233-797-1/pdf/hermode.pdf

allowable in the system's frequency ratios. In contrast, this algorithm takes as its model the complete harmonic series[6] with no limit on *n*, interpreting all input pitches as integer partials of a single fundamental. It then tunes each pitch according to the tuning of that partial in the harmonic series. Though this may not be classified as a proper *adaptive tuning* algorithm because it does not attempt to match a tuning system, this algorithm theoretically produces the most consonant results out of every algorithm described in this paper, as the resulting pitches are all integer multiples of a fundamental pitch.

Formally, this algorithm analyzes a set of frequencies A, producing a corresponding set of integer partial numbers P representing the lowest subset of the harmonic series whose intervallic structure closely matches that of A. In addition, it calculates f, the "virtual" fundamental frequency of this harmonic series. A parameter ε is supplied determining the allowable distance, or error, (in semitones) between each element in A and the nearby partial it is matched to. The following section explains the algorithm; pseudocode is provided in the Appendix.

A partial-finding algorithm

The algorithm first finds $a_{\min}$, the minimum value in A, and calculates the frequency ratio $r_k = \frac{a_k}{a_{\min}}$ for every other $a_k \in A$.

It then assumes the lowest pitch, $a_{\min}$, is the ith partial of the fundamental $f = \frac{a_{\min}}{i}$, with an initial value of $i = 1$. Upon each iteration of the algorithm, i is increased, consequently decreasing f.

To calculate the partial number p_k of each a_k, we can first represent the decimal r_k as a fraction, $\frac{d_{ik}}{i}$, the ratio between the "true" (non-integer) partial number d_{ik} of r_k, and $a_{\min}$'s assumed partial number i. We then calculate the error ratio e_{ik} between the true ratio $r_k = \frac{d_{ik}}{i}$ and the nearby integer ratio $\frac{\text{round}(d_{ik})}{i}$.

$$e_{ik} = \frac{\frac{\text{round}(d_{ik})}{i}}{\frac{d_{ik}}{i}} = \frac{\text{round}(d_{ik})}{d_{ik}}$$

If e_{ik} is within the pitch interval bounded by $\pm\varepsilon$, $\text{round}(d_{ik})$ is determined to be a potential partial number match for a_k. To verify this comparison, the frequency ratio e_{ik} is converted to its interval distance in pitch space:

$$e_{ik_{\text{pitch}}} = 12\log_2(e_{ik})$$

If $\left|e_{ik_{\text{pitch}}}\right| < \varepsilon$, $\text{round}(d_{ik})$ is set as the value of p_k and the process continues with a_{k+1}. Otherwise, the current values of i and f do not correctly represent the pitch collection—the algorithm increments i and tests it against every value in A once again.

Once all values of P have been determined, the analysis part of the algorithm is complete. To obtain a set of retuned frequencies A', it is only necessary to calculate $a_k{}' = p_k f$, where f is the most recently assumed fundamental.

Though there is no prescribed upper bound for the value of i, this algorithm is guaranteed to terminate, as the pitch space between partials in a given range decreases as f decreases.[7] For a standard value of ε

[6] In this paper, "harmonic series" refers to the musical harmonic series, defined as $f_n = nf_1$, where f_n is the nth partial (measured as a frequency) and f_1 is referred to as the fundamental or first partial.
[7] While in frequency space calculation of harmonic partials is a linear equation, in pitch space it is logarithmic, represented by the function $h_n = h_1 + 12\log_2 n$, where pitch h_n is analogous to the frequency f_n. The algorithm terminates because the derivative of the function (with respect to the partial number) is decreasing.

corresponding to a quarter tone[8] in pitch space ($\varepsilon = 0.5$), rounding the partials to the nearest integer enumerates the equal-tempered pitch classes that may be represented by the harmonic series. Figure 1 shows that all pitch classes are represented above an index of 14.

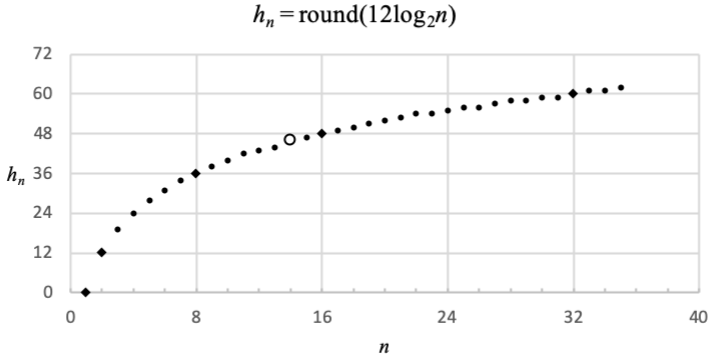

Figure 1. A graph of rounded partials h_n generated from an arbitrary fundamental of $h_1 = 0$. Above $n = 14$, shown by an open circle, every integer pitch in the octave (and therefore, every equal-tempered pitch class) is accounted for.

Pitch collections as graphs

This section introduces an interval-based adaptive tuning algorithm that imagines a given pitch collection as a complete graph, with vertices V corresponding to the pitches and edges E corresponding to all possible $\binom{|V|}{2}$ intervals between pairs of pitches. To adaptively tune the collection, a *spanning tree* is generated (a spanning tree is a connected subset of the graph with exactly $|V| - 1$ edges). The complete graph for the diatonic scale and a possible spanning tree are shown in Figure 2. Like the first algorithm, rather than generating a best-fit *temperament* for a given pitch collection, this algorithm selectively connects pitches only by pure intervals, generating a network that is locally and globally purely tuned.

A spanning tree is a fitting way to represent interval connections between pitches—with any fewer edges, the graph would not be connected. Likewise, with any additional edges, the graph would contain cycles, potentially leading to impossible structures such as the augmented triad in Figure 3. In fact, cycles in these graphs create impossibilities unless the product of their edges is a power of two (an octave).

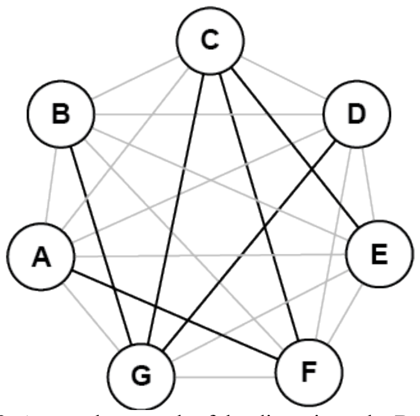

Figure 2. A complete graph of the diatonic scale. Bold black lines represent a possible spanning tree containing the intervals (edges) that are selected to be purely tuned in five-limit JI.

This representation closely matches interval structures such as pitch lattices used by Ben

[8] The quarter tone interval is half of a semitone.

Johnston[9] or the neo-Riemannian *Tonnetz,*[10] In 5-limit JI, the spanning tree of a diatonic scale may be represented by a subset of a two-dimensional pitch lattice, such as in Figure 4. This representation has further advantages. Though the spanning tree is generated in polynomial time, $\theta(n^3)$, in practical use its results can be calculated from a pitch collection in linear time, as explained in the section "Improving performance ".

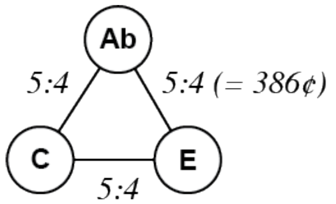

Figure 3. A mathematically impossible pure tuning of an augmented triad in a cyclic graph. Progressing once around the cycle from C back to itself, the interval C-C would be 125:64, which contradicts the assumption that every pitch will be exactly an octave (2:1) away from those within its pitch class.

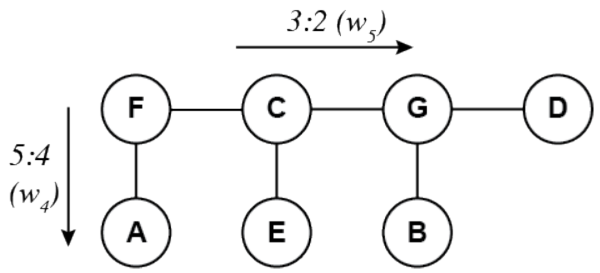

Figure 4. An equivalent spanning tree to Figure 2, arranged as a subset of the 5-limit JI pitch lattice. All following figures in this paper will be arranged similarly, with 5:4 and 3:2 ratios on the vertical and horizontal axes, respectively.

Modeling 5-limit JI with spanning trees

Table 1. Equivalences between interval class, frequency ratio, and diatonic interval in five-limit JI. m = minor, M = major, P = perfect, d = diminished, a = augmented.

Interval class n	5-limit JI ratio r_n	Diatonic interval names
1	16:15	m2 M7
2	9:8	M2 m7
3	6:5	m3 M6
4	5:4	M3 m7
5	3:2	P4 P5
6	45:32	a4 d5 (tritone)

In 5-limit JI, intervals commonly take on certain ratios, and these are collected in a set R which is indexed by interval class[11] $n = 1, 2, 3 \dots 6$. These are shown in Table 1—note that the ratios occasionally correspond to an interval's complement (whose ratio is its inverse) rather than the interval itself.

Each interval (edge) in the spanning tree must be assigned a weight in accordance to its importance within 5-limit JI (the most desired intervals will have the least weight). Octave equivalence is assumed, so we define a set of weights W corresponding to the interval classes; a system of linear inequalities helps to enforce some rules observed from the JI diatonic scale. The following rules, encoded in the values of W, are used by the algorithm to construct paths in the spanning tree:

1. $w_5 < w_4 < w_3 < w_2 < w_1 < w_6$

[9] Fonville, John. "Ben Johnston's Extended Just Intonation: A Guide for Interpreters." *Perspectives of New Music*, vol. 29, no. 2, 1991, pp. 106–137. www.jstor.org/stable/833435.
[10] Tymoczko, Dmitri. "The Generalized Tonnetz." *Journal of Music Theory*, vol. 56, no. 1, 2012, pp. 1–52. www.jstor.org/stable/41508604.
[11] The term *interval class* used here is a term borrowed from post-tonal music theory, defined as a value from 1 to 6 that counts the semitones between two pitches "mod-12". Mod-12 complements are considered equivalent, so the interval class 5 represents both the interval of 5 semitones (a perfect fourth in tonal theory) and that of 7 semitones (a perfect fifth). Table 1 is provided to clarify terminology.

- A fifth is the most desired interval, followed by a major third, then a minor third, etc.

2. $w_4 < 4w_5$
   - It is better to construct a major third directly, as a 5:4 ratio, than to construct it by stacking four fifths, as an 81:64 ratio.
3. $w_4 + w_5 < 3w_5$; therefore $w_4 < 2w_5$
   - It is better to construct a major sixth with a fourth and a third, as a 5:6 ratio, than to construct it with three fifths, as a 27:16 ratio. This supersedes the previous inequality.
4. Fifths and thirds must be used whenever possible. Therefore, the weights $w_3, w_2, w_1,$ and w_6 must be large enough to only be chosen when no possible paths include fifths or thirds.

A possible set of weights in five-limit JI is $W = \{10001, 1001, 100, 2, 1.1, 100001\}$. The larger values chosen are relatively arbitrary.

An intuitive approach from this point would be to generate a *minimum spanning tree* (MST), the spanning tree with the minimum total weight. The MST approach may appear to likewise reduce dissonance, but this approach does not produce an accurate representation of the JI scale. For a diatonic scale, the MST algorithm would "greedily" pick this cheapest option, w_5, at every step, forming a stack of pure fifths, or the Pythagorean scale, no matter what weights are assigned to the other intervals. In addition to compromising pure major thirds, the resulting tree is a straight line, so the path between the farthest vertices, or "diameter" d, is rather large, $6w_5$. The frequency ratio between the farthest vertices, the augmented fourth, is quite complex: $(\frac{3}{2})^6 = \frac{729}{64}$.

A better solution would be a balance between minimum total weight and minimum path length between any two vertices. A fitting solution is found by generating a *minimum diameter spanning tree* (MDST),[12] which minimizes the length (weight) of the longest path, in turn keeping all vertices as close as possible to a "center" which may be either a vertex or a pair of vertices.

Once the spanning tree has been generated, the pitches may be retuned by the following method: first, choose a starting vertex v_0 from the tree and record its frequency y_0. Then traverse the tree from that vertex and assign to each other vertex v_i a frequency y_i which is the product of y_0 and all frequency ratios $r \in P_i$, where P_i is the path containing all edges from v_0 to v_i. Because r sometimes represents the inverse of the interval (i.e. 2:3 rather than 3:2), it is important to take the inverse of ratios when traveling "right to left" on the lattice.

$$y_i = y_0 \prod_{r \in P_i} r$$

Each traversal of a tree visits the same edges, so any starting vertex produces the same collection shifted up or down in pitch. For consistency, a certain pitch, such as the lowest, could be always chosen as the reference pitch. Alternatively, the collection could be shifted according to the tuning of the previously calculated collection(s)—see the section "Adaptive tuning with respect to time".

Performance of the MDST algorithm

With the given weights prescribed in the previous section, the algorithm generates

[12] Ho, Jan-Ming, et al. "Minimum Diameter Spanning Trees and Related Problems." *SIAM Journal on Computing*, vol. 20, pp. 987-997, 1991.

spanning trees matching those of the 5-limit JI diatonic and chromatic scales,[13] shown in Figure 5 and Figure 6 as collections based on C.

In addition, this model also represents simultaneous, discrete triads in their most pure form, enabling justly-tuned polytonality. The examples shown are two triads a tritone apart (Figure 7) and a hexatonic collection containing three triads (Figure 8); in both cases, the MDST algorithm tunes the triads as purely as possible.

Though the examples shown in this paper are limited to 5-limit JI, this algorithm could be applied to higher limits by adjusting the interval weights W and the frequency ratios R.

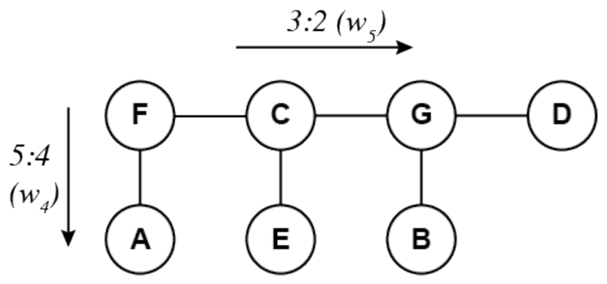

Figure 5. Generated MDST of a diatonic scale in 5-limit JI (identical to Figure 4). The diameter in this case is $d = 3w_5 + w_4$, the path from A to D.

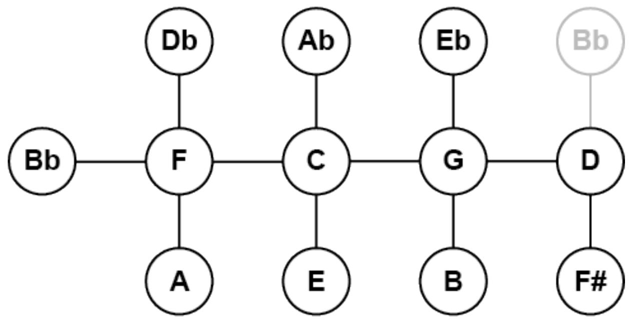

Figure 6. Generated MDST of a chromatic scale in 5-limit JI, where $d = 4w_5 + w_4$ (Bb to F#). Another common JI tuning of the Bb is shown in grey, more symmetrical but with a resulting larger diameter $d = 3w_5 + 2w_4$ (grey Bb to A).

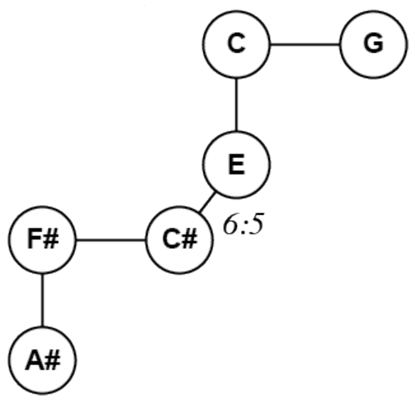

Figure 7. MDST connecting two triads a tritone apart; $d = 2w_5 + 2w_4 + w_3$.

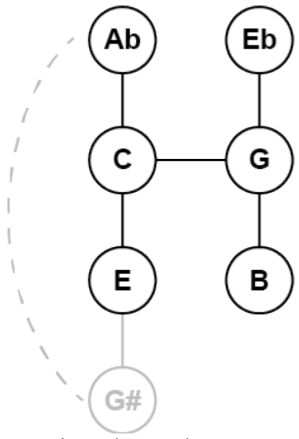

Figure 8. A hexatonic scale tuned as two augmented chords a fifth apart. Justly tuning all three triads (C Major, E Major, and Ab Major) would require the bottom G# to be equal in pitch to the top Ab, an impossibility shown in Figure 3. However, the MDST algorithm finds the most fitting solution, pictured in bold black lines: $d = w_5 + 2w_4$ (Ab to B).

Improving performance by storing pitch class set tables

The calculation of the MDST makes certain assumptions about intervals:

1. A pitch class will always be tuned the same in any octave
2. An interval a is considered equivalent to its complement $12 - a$, as well as any compound interval $a + 12x$, $x \in \mathbb{Z}$.

[13] Note that these scales are complex examples, or "edge cases", that would not commonly occur in practical use. Obviously, this model performs well on common pitch collections as well.

These assumptions lead to the interpretation of pitches as pitch classes and intervals as interval classes. Because of the invertible nature of interval classes, an isomorphic spanning tree will be generated for any transposition or inversion of a pitch set.

With these assumptions, the number of possible spanning trees is greatly reduced, and in fact it is possible to represent any possible pitch class set by its prime form, of which there are only 224.[14] Because there is only one possible interval to be formed in a set of two notes, it is not necessary to store either those sets nor the zero- or one- element sets, which reduces the number to 216.

Because of the small number of possible tuning schemes, it is practical to create a table of schemes indexed by their pitch class set. The actual adaptive tuning algorithm need only be calculated once for a given set of parameters (such as interval weights in the case of the MDST). The MDST algorithm is computationally complex in terms of the number of steps, in the order of n^3 for input size n, but to access the precalculated table only requires a calculation of prime form, one that can be completed in linear time (in the order of n) using an algorithm that represents pitch class sets as binary strings (integers).[15]

Adaptive tuning with respect to time

This paper has only discussed adaptive tuning of simultaneous pitch collections. Though not the focus of the paper, it is necessary to briefly mention "horizontal" tuning. As discussed by Stange et al.,[16] intonational differences are not only perceived simultaneously, but over time as well (pitch memory is strongest for a period of about three seconds). For this reason, discrete collections, whether spanning trees or sets of partials, should be connected in a way that minimizes the total change in tuning of all pitch classes shared between the collections. In the case of both models presented in this paper, this adjustment would be an operation that shifts every pitch of a collection slightly up or down.

While this paper contributes no comparable solution to "horizontal adaptive tuning" techniques proposed in existing papers such as those by Mohrlok or Stange et al., horizontal tuning may be best left to the user of the implementation:

Just intonation presents the potential for microtonal modulation, or "drift" over time. When creating an adaptive tuning system, this type of behavior would likely be avoided in favor of staying relatively near a center

[14] The prime form of a pitch class set is its transposition or inversion which is most "densely-packed" from the lowest pitch upwards. The number of prime forms is supplied from Rahn:
Rahn, John, *Basic Atonal Theory*. New York and London, Longman, Inc., 1980.

[15] This algorithm is from the code for an Android app:
Rose, Nick (nihk). "Pitch Class Set Calculator." *Github*, 2016. https://github.com/nihk/Pitch-Class-Set-Calculator/blob/master/app/src/main/java/com/nihk/github/pcsetcalculator/utils/SetTheoryUtils.java

[16] Stange, Karolin, et al. "Playing Music in Just Intonation: A Dynamically Adaptive Tuning Scheme." *Computer Music Journal*, vol. 42, no. 3, 2018, pp. 47-62. https://www.mitpressjournals.org/doi/pdf/10.1162/comj_a_00478

pitch.[17] However, musical possibilities afforded by harnessing drift may be found in the music of composers such as Ben Johnston and Kate Soper, as well as in popular music such as the vocal arrangements of Jacob Collier.[18] Here, the conflict between the "goal" of a black box adaptive tuning algorithm and this music (which is composed using a similar model) offers evidence that the potential of adaptive tuning models is maximized when they are made interactive, especially in an implementation that includes a temporal element, such as a digital musical score.

Conclusion

The algorithms presented in this paper retune pitch collections in real time, accurately recreating the models of just intonation and the harmonic series. As is the case with any adaptive tuning algorithm, they make tuning choices so that the user (performer) does not have to. These algorithms perform in a way that not only produces expected results, even for complex pitch collections, but does so efficiently. However, adaptive tuning systems such as these realize their full potential only when they are implemented as an interactive model for music composition. The algorithms presented in this paper offer an intuitive simplicity that makes them prime candidates for such an implementation.

Appendix

Pseudocode for the "partial-finding" algorithm

Inputs: A, a set of pitches measured in frequency, and ε, a fractional pitch value
Outputs: f, the fundamental frequency, and P, a set of integer partial numbers

```
function findPartials:

Find the minimum pitch, $a_{\min} \in A$.

For every other $a_k \in A$:
    Calculate the frequency ratio from the lowest note
    $r_k = \frac{a_k}{a_{\min}}$
```

[17] This is accomplished effectively by Hermode Tuning, which coaxes pitches back to a center reference pitch when the system begins to drift.
[18] Listening examples include the following:
Ben Johnston – String Quartet No. 7, III.
Kate Soper – Nadja
Jacob Collier—In the Bleak Midwinter

```
While not all entries of $P$ have been determined:
  *Assume $a_{min}$ is the $i$th partial of $f$*
  $p_{\min} = i$
  $f = \frac{a_{\min}}{i}$
  For all $r_k$:
    *Calculate the denominator $d$ of the ratio $r_k$*
    $d = r_k i$
    if $\left|12 \log_2 \left(\frac{\text{round}(d)}{d}\right)\right| < \varepsilon$
      $p_k = \text{round}(d)$
    else
      *Try again with a greater value of $i$*
      break

Return $f$ and $P$
```

References

Sethares, William A. *Tuning, Timbre, Spectrum, Scale*. 2nd ed., London, Springer-Verlag, 2005, pp. 162-178.

Mohrlok, Walter. "The Hermode Tuning System." Self-published, 2003. http://extras.springer.com/2005/978-1-85233-797-1/pdf/hermode.pdf

Fonville, John. "Ben Johnston's Extended Just Intonation: A Guide for Interpreters." *Perspectives of New Music*, vol. 29, no. 2, 1991, pp. 106–137. www.jstor.org/stable/833435.

Tymoczko, Dmitri. "The Generalized Tonnetz." *Journal of Music Theory*, vol. 56, no. 1, 2012, pp. 1–52. www.jstor.org/stable/41508604.

Ho, Jan-Ming, et al. "Minimum Diameter Spanning Trees and Related Problems." *SIAM Journal on Computing*, vol. 20, pp. 987-997, 1991.

Rose, Nick (nihk), "Pitch Class Set Calculator." *Github*, 2016. https://github.com/nihk/Pitch-Class-Set-Calculator/blob/master/app/src/main/java/com/nihk/github/pcsetcalculator/utils/SetTheoryUtils.java

Stange, Karolin, et al. "Playing Music in Just Intonation: A Dynamically Adaptive Tuning Scheme." *Computer Music Journal*, vol. 42, no. 3, 2018, pp. 47-62. https://www.mitpressjournals.org/doi/pdf/10.1162/comj_a_00478