The Rationale behind Rationale: Designing a Sequencer for Unlimited Just Intonation

Chuckk HUBBARD
badmuthahubbard.com
Bucharest, Romania
chuckk.hubbard@gmail.com

Abstract

This article presents some of the considerations that went into determining how the Rationale Just Intonation sequencer should work. Various special problems that came about because of the number and nature of usable tones are discussed, as well as reasons for eschewing other existing notation systems. Programming specifics are ignored in favor of questions of interface design.

Keywords

Rationale, Just Intonation, xenharmonic, microtonal, composition

1 Introduction

Among the many things made vastly simpler by computers is the generation of tones of very precise arbitrary frequency. That should mean that composing in xenharmonic[1] tunings, i.e., unusual tunings, is now a piece of cake. It is, if the composer knows exactly what tones should be sounded and when, which is to say, if the piece has already been composed, away from the computer. Otherwise, there are text-based scores to edit manually and then process (e.g. traditional Csound score format, Scala's sequence file format[2]), or there are sets of predetermined possible tones that can be chosen from (e.g. most retunable soft or hard synths triggered by MIDI). Most existing options use one of these two methods to bridge the gap between the composer's creativity and the computer's power, i.e., to input notes.

There's nothing wrong with composing using text files; it can and does produce brilliant results. Still, it can only be a good thing for there to be more options, and for many people, seeing and dragging notes is more intuitive. As for sets of predetermined possible tones, there is theoretically no limit to the number of tones that could occur in Just Intonation, but the relationships between them are very important, so having too many of them available at once is almost as bewildering as having too few. You could set up a software synthesizer with thousands of tones per octave, but this would make it harder, not easier, to intuitively choose the right tone.

Just Intonation is a method of musical tuning whereby all frequencies have whole-number ratios[3] with all of the others. The simplest such ratio, which is the only one unchanged in the majority of tuning systems, is a ratio of 2 to 1, commonly called an octave. A tone with a frequency of 300 Hertz will sound an octave higher than a tone with 150 Hertz. A ratio of 3:2 is what we call a perfect fifth, e.g. a tone of 300 Hertz compared to a tone of 200. A ratio of 5:4 is a major third, but in 12-tone equal temperament, this interval is already noticeably detuned. Three tones in the ratio 4:5:6 will sound as a major triad, e.g., C E G.

Simple music using these tunings will probably not sound especially groundbreaking to most Western listeners. We have a strong innate tendency to sing simple harmonies and melodies in Just Intonation, even when struggling to sing in tempered tunings. But, a few hundred years ago, some composers began to crave more harmonic freedom to change keys quickly, without going out of tune, and using a small number of total notes-around the time of the advent of the keyboard. This last requirement excluded Just Intonation. Three successive major thirds in Just Intonation would go from 5:4 (C-E) to (5x5):(4x4) (C-G#) to

1 While the term 'microtonal' would be more immediately clear to many people, it has become irksome to most enthusiasts, since much of the music composed in xenharmonic tunings doesn't use microtones (smaller than semitones) at all. They sometimes occur in Just Intonation, but the music can hardly be defined by this element; the *large* uncommon intervals are just as important.

2 Scala's sequence file format: http://www.huygens-fokker.org/scala/seq_format.html

3 I.e., the ratio of any positive integer to any other positive integer.

(5x5x5):(4x4x4) (C-??[4]), that is, 125:64. An octave is 2:1, or 128:64, and no one wants to build accordions with separate keys for 128:64 and 125:64 and all of the possibilities in between. It was simpler to alter that 5:4 ratio so that applying it three times would give exactly an octave. By adjusting some of the intervals, the number of possibilities was deliberately limited for practicality.

Rationale[5] is a flexible sequencer for composing in extended Just Intonation, first released in 2008. It is free, licensed under GPL v.3, and depends on the Python interpreter, the Python Tkinter toolkit (included by default with the Python interpreter), and the Csound API for Python. It is theoretically cross-platform, but has only been tested extensively on Linux. The idea with Rationale was to have a graphical score on which notes can be placed that have certain frequency ratios to other notes, but that, at any time, the composer may wish to modulate, permanently or temporarily, or simply to see the relations between any arbitrary set of notes, not always including the starting tonic. The ratios of 3:2, 5:4, 6:5, and 7:4 are all options in most JI systems, but if one wants to make that 7:4 into the tonic of a new harmony, it may be necessary to include ratios like 49:32 (7:4 x 7:8), 63:32 (7:4 x 9:8), and so on. With experience, the meanings of those ratios become more obvious too, but there are no limits; some composers may be fine with using a ratio like (7^5):(2^5*3*5^3), i.e., 16807:12000, but again, more options is a good thing, and some composers will prefer something more intuitive.

Ben Johnston's system of accidentals, among others, is one way of making this more intuitive. No written ratios are needed. He starts with a diatonic C major scale and uses various symbols to notate specific adjustments. He uses the standard ♭ and ♯ familiar to most musicians, as well as + and -, ↑ and ↓, and small superscript numbers 7, 13, 17, and further prime numbers with their upside-down counterparts, to specify these adjustments next to standard notes on a staff.[6] He has composed extraordinary music this way, as have many others, but to understand the scores takes serious study; rows of accidentals may be placed in front of any one note to show how it is related to other notes with almost as many accidentals. It is perfectly usable, but I was looking for something else.

Harry Partch wrote that, "It is quite conceivable that an instrument could be built that would be capable of an automatic change of pitch throughout its entire range, up or down by any reasonable interval, and if Just Intonation can surmount the many hazards and problems ... the problem of transposition may be considered minor, one for which a solution will inevitably be found."[1] He didn't seem to devote much energy to inventing such a retunable instrument, preferring tuned reeds, fixed-length strings and stuck percussion. If I had lived at a time when computers didn't exist or were very expensive, I probably wouldn't have tried to create such an instrument either. But that comment of his seems very compelling at a time when GUI programming is as advanced as it is today.

My decision to create a Just Intonation sequencer with automatic tonality changes, without accidentals and with an undetermined number of tones led me to a series of unusual decisions during the creation of Rationale.

2 Automatic tonality changes

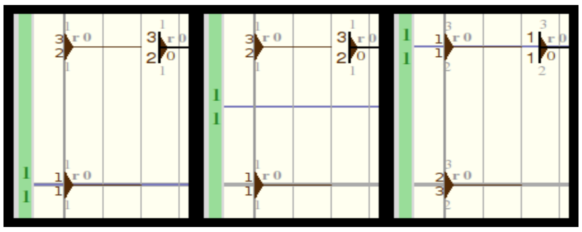


Figure 1: Before, during and after a tonality change. The two forms on the left in each frame are entered notes and the one on the right is the hover.

By "automatic tonality changes", I actually mean redrawing existing notes with different reference points, but the same frequencies. The notes on the screen do not move, but the ratios change and the horizontal reference lines slide to new positions. In traditional theory, this could be compared to transposing by, perhaps, a diminished second: same sound, different note names. This is useful for several reasons. A composer may want to modulate (multiply) repeatedly during a piece, and, as mentioned, if you modulate e.g. by a major third three times, you arrive at 125:128, not 2:1. With Rationale, I wanted to be able to always be looking at small-number intervals. This "modulation" could also be very temporary, just for some short phrase to be repeated at different frequencies before returning to a broader theme- something that is a piece of cake in traditional notation.

There is a kind of secondary mouse cursor used in Rationale for placing notes. As the mouse moves, it cycles through a set of ratios, and a click places a note at the appropriate height. In the

4 This would be 41 cents below C, or 41/100ths of a semitone.

5 Rationale : rationale.sourceforge.net

6 http://en.wikipedia.org/wiki/Just_intonation#Staff_notation

documentation, this cursor is referred to as the *hover*. The hover has another important purpose, which is to select the ratio that will become 1:1 (the tonic, representing a 1:1 frequency ratio with the reference frequency) after a tonality change. The tonality change is accomplished by hitting 'T' on the keyboard when the hover is at the desired ratio. All existing notes' ratios change and the hover's ratio becomes 1:1, wherever it is. Afterwards, the hover can be moved to a new ratio and another tonality change performed, again and again. This idea was evidently subconsciously inspired by exposure to the concept of *movable do*.[7] You can always return to the original 1:1, which is by default middle C.

2.1 Tonality regions

The problem that arises when jumping arbitrarily from *do* to *do* is that the earlier notes entered with simple ratios often show more and more complex ratios as *do* changes. The computer doesn't care, but the composer should be able to see the simple relationships in different "keys" easily; not all notes need to change. The solution was tonality regions. Holding down 'R' and typing a number will switch the hover to that region; future tonality changes will only affect notes in that region. Every note shows its region next to a small letter 'r'. This region symbol can be assigned different colors and shades to be more visibly distinct.

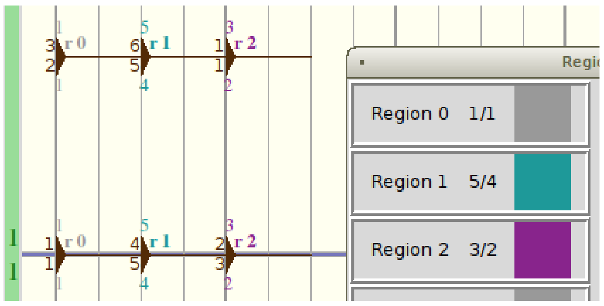


Figure 2: Three different regions with the region dialog. The entered notes sound exactly the same.

3 Decisions influenced by the number of tones

Many decisions were influenced by the need to differentiate between a large number of tones within a small space. As mentioned, others have added more variations of accidentals to traditional notation. In turning away from accidentals, Rationale cannot fit an octave into one inch, with five lines and four spaces. Even with the ratio for each note displayed by the note, the resolution of the screen would force some ratios to be skipped while moving the hover.

3.1 Universal staff

It is extremely unwieldy to show each instrument on a separate staff when each instrument requires a large amount of screen space. The concept of a staff is thus almost completely absent from Rationale. Horizontal guide lines are present but are all one octave apart. There is one extra horizontal guideline that always remains at middle C while the rest move according to the current position of 1:1. All of the instruments are represented in the same area and frequency is vertically absolute, i.e., a vertically higher note always has a higher frequency (although what that frequency actually means depends on the design of the instrument). In very simple songs, this causes no problems, but in more involved compositions, notes from different instruments are hopelessly mixed around each other, even superimposed over each other. The benefits of a universal staff are that one can see everything that is happening and it is really quite intuitive; the instruments are not isolated from each other. And, of course, there is room to visually differentiate between hundreds of possible frequencies. I didn't invent the idea; some of Stockhausen's famous scores, e.g., represent sound the same way, albeit without printed frequency ratios. Rationale has several features intended to resolve various problems presented by this universal score.

3.1.1 Shortcut for switching instruments

In typical sequencers, the instrument being entered depends on which staff the mouse approaches. That's impossible with only one staff, so the next most efficient way is to switch instruments with a keyboard shortcut. That shortcut is to hold Shift while typing a number. Rationale can easily provide over 1,000 instruments.

3.1.2 Different colors for instruments

The most obvious need with a universal staff is simply to see which instrument a note belongs to. Each instrument starts medium gray, and colors can be assigned to them arbitrarily. Typically, various instruments are all assigned sharply contrasting colors, but it is up to the composer. The structure of a piece can be easily visible with such a setup. So instead of different instruments'

7 Where, upon modulating to a new key, the tonic of the new key is treated as *do* in solfeggio, rather than saying, for instance, that you have modulated to *re*.

notes looking the same but being in different places, they are all in the same place and look different.

3.1.3 Independent horizontal and vertical zoom

Rationale also uses proportional notation, meaning the duration of a note is represented by its physical length, but this was more a matter of simplicity than a major decision. Still, with notes arbitrarily close to each other, it was necessary to be able to manually change the vertical scale without affecting the horizontal, and vice versa. Horizontal zoom is changed with -/+, zooming in and out respectively, and with Backspace, which resets the zoom to default. Vertical zoom is controlled the same way but while holding the Shift key. Such separate zoom control is surprisingly useful. It amounts to more control over how you see your music.

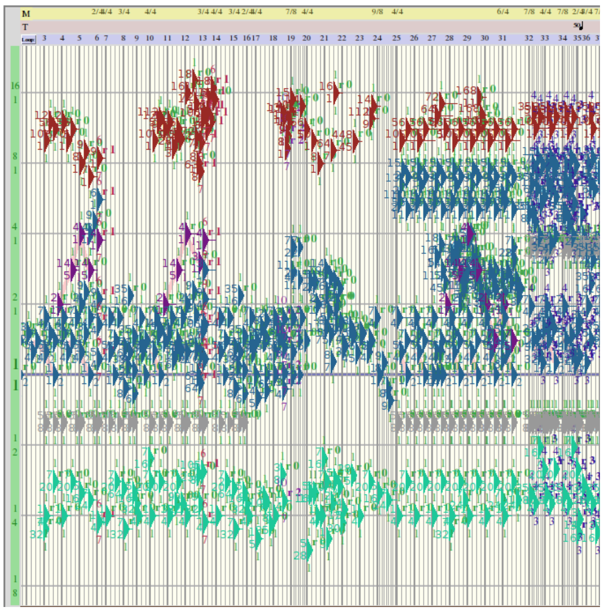


Figure 3: The universal staff, zoomed out both horizontally and vertically to show structure

3.1.4 Mutually exclusive horizontal and vertical note dragging

Rationale snaps notes to a customizable rhythmic grid, but the vertical grid, representing frequency, is normally much finer: at times, a movement of one pixel changes a note's ratio.[8] It would be easy to accidentally change frequency when trying to change time. For this reason, clicking on a note in EDIT mode only allows it to be dragged vertically. Holding the Shift key and dragging the note only moves it horizontally. This allows broad mouse gestures for moving notes drastically without changing their ratios, or without changing their times. Large groups of notes can be moved this way.

3.1.5 Hiding / showing specific instruments

Sometimes zooming in isn't enough to make all the notes clearly visible. It wasn't until I had worked with a fully functional Rationale for a while that I realized that two or more instruments might have to play the same frequency at the same time, and that, with the then-current incarnation of the program, which instrument's note ended up visible (and editable) was arbitrary. Being able to instantly hide all notes from any instrument seemed the best solution available. This is achieved by holding the Alt key and typing the number of the instrument. If an instrument is hidden when selecting a broad area, its notes are not selected. However, if previously selected notes are hidden, they remain selected and are still affected by any operations performed while they are hidden. The Alt key + the instrument number will show that instrument again, and Alt + 'S' will show all instruments. The only other way I could think of to manage superimposed notes of different instruments would have been with a three-dimensional score, which is beyond my programming abilities.

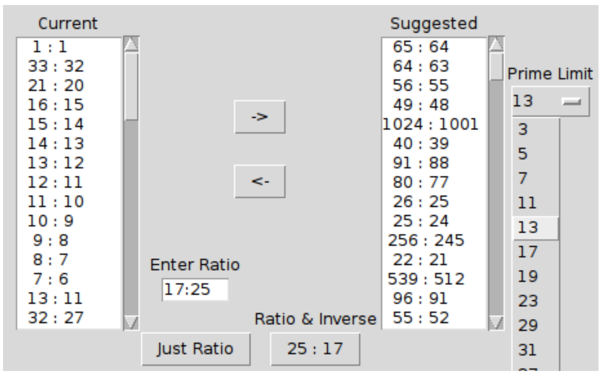
3.2 Notebanks

It was mentioned in the introduction that Rationale is an alternative to using accidentals, but also an alternative to finite, predetermined sets of tones. Part of the escape from predetermined sets is in the arbitrary tonality changes, and part of it is in notebanks. The default set of frequency ratios through which the hover will cycle has 57 possibilities in an octave, but this can be customized easily, and separate banks of ratios can also be remembered and switched between. This also allows, for instance, having higher prime-limit ratios set aside for when needed, or simply dividing a set of ratios into multiple smaller sets. For example, One's default notebank could include ratios like 1:1, 9:8, 5:4, 3:2 and 7:4, and a separate notebank could have higher harmonics like 17:16, 19:16 and so on. This is one way to have less notes available at any moment, allowing one to zoom out more vertically. Regardless, the idea is that there is no predetermined set of possible tones. The only limit is that the ratios must use only positive integers.

Holding the Control key and typing 'B' will open the notebank dialog. From there, notes can be

[8] The horizontal rhythmic grid is, as mentioned, customizable, and quantization could be set to 1/9999 of a quarter note, for example, so it could be made finer than the vertical frequency grid if desired.

traded in and out of the chosen notebank, and more notebanks can be created. It is possible to specify here arbitrary whole-number ratios. When finished, it is possible to switch between active notebanks by holding 'B' and typing a number. The hover will then cycle through the ratios listed in the active notebank until it changes again.



Current
1 : 1
33 : 32
21 : 20
16 : 15
15 : 14
14 : 13
13 : 12
12 : 11
11 : 10
10 : 9
9 : 8
8 : 7
7 : 6
13 : 11
32 : 27

Suggested
65 : 64
64 : 63
56 : 55
49 : 48
1024 : 1001
40 : 39
91 : 88
80 : 77
26 : 25
25 : 24
256 : 245
22 : 21
539 : 512
96 : 91
55 : 52

Prime Limit
13
3
5
7
11
13
17
19
23
29
31

Enter Ratio
17:25
Ratio & Inverse
25 : 17

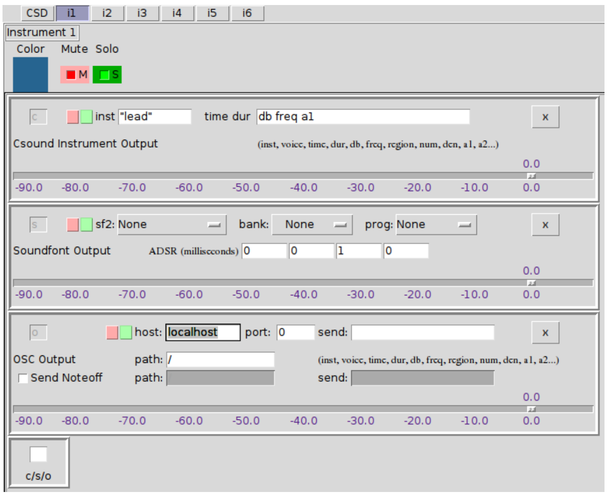
Figure 4: The notebank dialog. If an arbitrary ratio is typed, there is an option to add just that ratio and an option to add that ratio and its inverse. All ratios will be adjusted to be between 1:1 and 2:1

Rationale maintains a list of mouse positions for the ratios in the notebank. If several ratios are very close to each other, Rationale will try to make them each fit somehow, in the sense that the mouse should not skip any of them as long as there are as many pixels within an octave as there are ratios in the active notebank. If you have 40 pixels in an octave and 20 ratios, but all very close to each other, the hover shows them farther apart than they really are so they can all be accessed, but entered notes will appear at the correct height.

4 Output options

Rationale has no ambitions of becoming an audio processing tool. The Csound engine is used for all audio and all timing. What to do with the note data once it is composed is basically left to the user, although it can be sent out in several forms.

The output dialog appears upon pressing 'O'. The first tab displayed is for loading or typing a Csound csd file. There is an option to automatically reload this file every time playback starts, which creates the possibility of looping a segment and editing the csd file in a text editor in the meantime.[9] A new tab is created in this dialog every time a new instrument is created. Each instrument's tab has the possibility to add numerous outputs, which can be Csound instruments, Soundfont programs, or OSC commands.[10] Any time an instrument is called, it will send messages to all outputs listed on its tab, unless they are muted. It is possible to create outputs for different combinations of Rationale instruments to Csound instruments, and to selectively mute them, in order to quickly switch parts between instruments.



Instrument 1
Csound Instrument Output
inst "lead"   time dur db freq a1
(inst, voice, time, dur, db, freq, region, num, den, a1, a2...)

Soundfont Output
sf2: None   bank: None   prog: None
ADSR (milliseconds) 0 0 1 0

OSC Output
host: localhost   port: 0   send:
path: /
Send Noteoff
(inst, voice, time, dur, db, freq, region, num, den, a1, a2...)

Figure 5: A Csound output, a Soundfont output and an OSC output.

Most of the elements of these outputs are standard for sequencers, with one exception: Csound and OSC outputs have message boxes to construct the messages to be sent with each note. Csound outputs by default send the time and duration of the note first, and after that any aspect of the note may be added in any order. Things like the instrument number, numerator, denominator, or region can be sent with each note. There may also be arbitrary fields, entered as a1, a2, a3, etc. These arbitrary fields can be assigned per note by right-clicking on the score. This allows fine control of Csound's many complex forms of synthesis. Granular synthesis, for example, needs much more information than the time, length, frequency and loudness of each note. The ability to draw automations for these values is a goal, but a distant one. OSC outputs have similar custom message-building, while Soundfont output has the usual Soundfont options.

5 Conclusion

One driving goal behind Rationale was to make the composition process in extended Just

[9] The mechanism of looping, like many others, is fairly standard among sequencers, and so not addressed in this paper.

[10] Unfortunately, MIDI output has proven too complicated so far. Many composers have asked about it, and it would open up many possibilities for sound production, but so far it has not materialized.

Intonation as intuitive as possible, in the hopes that more composers, who may not have initially been curious, would be drawn to experiment with the tuning system. The kinds of gestures they use in their other works or on their own instruments should be relatively simple to achieve in Rationale. This paper should have made the decisions involved in creating that environment a little clearer.

Using Rationale involves many other features, but most of them are more or less standard. This paper addresses mainly features that resulted specifically from the demands of composing in unlimited Just Intonation, namely the large number of tones and the inclusion of all instruments on one staff. Selecting notes, copy/paste, saving, undo/redo, transport, meter and tempo changes, note loudness, looping, audio options and the four basic modes (ADD, EDIT, DELETE and SCRUB) are all present but are not specific to this scope. They are documented in the Rationale help file.

6 Acknowledgements

Many thanks to all of the Csound developers, and in particular Victor Lazzarini, John ffitch, Oeyvind Brandtsegg, Rory Walsh, Michael Gogins and Steven Yi. All have explained things to me and/or fixed things for me over the course of several years, asking nothing in return. Thanks also go out to the xenharmonic community, including Dr. John Chalmers, Aaron Krister Johnson, Carl Lumma, Toby Twining and many others.

I must also thank my wife Irina for her love and support and for not throwing my computer in front of a subway train while I was programming Rationale.

References

[1] H. Partch. 1974. Genesis of a Music. Da Capo Press, New York, New York.