Returning to the Fundamentals on Temperament (In Digital Systems)

Nathan Renney
Computer Science Research Centre
University of West England
Bristol, United Kingdom
nathan.renney@uwe.ac.uk

Benedict R. Gaster
Computer Science Research Centre
University of West England
Bristol, United Kingdom
benedict.gaster@uwe.ac.uk

Tom Mitchell
Creative Technologies Lab
University of West England
Bristol, United Kingdom
tom.mitchell@uwe.ac.uk

ABSTRACT

Considering the generation of musical tunings, it is reasonable to expect that the many constructs contained in Functional programming languages may provide useful tools for exploring both conventional and new tunings. In this paper we present a number of approaches for manipulating tunings using basic mathematics. While this provides a simple foundation for describing temperament, it is fundamental enough to support a variety of approaches and further, allows the unbounded description of arbitrary tunings. It is expected that this notion will be useful in defining tunings, and by extension scales, for Digital Musical Instruments. This breaks down the physical barrier that has limited the likes of just intonations from having practical applications in the performance setting. It also enables composers to explore a variety of non traditional temperaments rapidly, without having to manually tune each note.

1 INTRODUCTION

Western classical tradition settled on the compromise of twelve tone equal temperament (12-TET) in the 19th Century and it has since become an overriding, although necessary, standard for composition and musical instrument design [2]. Many instruments are physically bound by an inability to tune or modulate to just intonations in a practical setting and as such, use of this form of harmony is constrained for most ensembles. The legacy of these physical restrictions mean it is uncommon for extensive support for alternative temperaments with discrete pitched instruments, digital controllers or software instruments. As such, composers typically meet many challenges in exploring this form of harmony in a manner that is intuitive or supports rapid, iterative experimentation. Furthermore, with 12-TET deeply ingrained in our notion of harmony many theoretically acceptable harmonic relations no longer have the same effective perceived quality in an alternative temperament. Even violinists, with continuous, fine grained control over pitch, struggle to effectively modulate between keys in just intonation, partly because of the technical challenges involved but also because the open strings of the instrument are tuned to a fixed pitch and cannot easily be adjusted in a performance setting.

For the most part, the details of tuning and temperament remains a hidden layer upon which western classical notation and theory sits. Given the mathematical foundations on which alternative systems of temperament are built and the ease these calculations can be performed on modern computational systems, we can expect that there is a way to describe different tunings in a way that is both clear and allows for accessible, creative exploration.

As the concept of a scale or tuning naturally fits the data structure of a list, a functional language such as Haskell [8] provides an excellent set of tools and concepts for implementing these ideas. This paper aims to examine the potential ways for existing tunings to be expressed and, beyond that, how functional programming may present a foundation for exploring new tunings and by extension, harmony. The results should allow scales to be created from a base tuning and ultimately applied within digital instruments.

A Domain Specific Language (DSL), such as those described by Hudak [6], provides a language for describing specialised programs, simplified to the constraints of a single problem space. We can presume a DSL that utilises the ideas from this paper would resemble the code fragment shown in figure 1, where a 4 by 4 grid is created and mapped onto a generated 12TET diminished blues scale. A related hardware controller could then be configured using the output from this program.

```
grid = Button 4 <+> Button 4
referencePitch = 440
freq_12tet_dimBlues =
  [ 2.0 ** (n / 12.0) * referencePitch | n <- diminishedBlues]
applyToGrid grid freq_12tet_dimBlues
```

Figure 1: A DSL to map tempered scales to interfaces

With this interface, a composer may experiment with a generated tuning in a way that resembles traditional instruments, having spent minimal time configuring individual note tunings.

This paper's key contribution is to lay the foundations for a reimagining of temperament in the context of digital instrument design and domain specific languages. This paper is structured as follows:

- Section 2 outlines the typical forms of temperament and tuning systems.
- Section 3 discusses some of the current interactions with arbitrarily tuned digital instruments and some better known examples of alternative temperaments in relatively modern composition.
- Section 4 presents the basis for mathematically generating tunings.
- Section 5 provides a brief, practical example of how functional language features can implement these concepts.
- Section 6 describes applications to which the concepts discussed in this paper could apply.
- Section 7 concludes, commenting on the future expansion of this work.

2 TUNING AND TEMPERAMENT

A comprehensive description of tuning systems is beyond the scope of this paper. We will introduce the most recognised types of temperament here and further reading is recommended to consider the subtle variations that exist within each.

There are a number of ways to describe the relationship between notes. Due to the ubiquity of twelve tone equal temperament (12TET), the most common unit tends to be cents, a logarithmic unit adopted and developed by Alexander J. Ellis for his work comparing tunings from around the world [3]. This unit sees an octave within 12-TET divided into a geometric sequence of 12 divisions, each equal to 100 cents, referred to as a semitone.The MIDI specification, along with the majority of compatible synthesisers, adopt this model where alternate temperaments are described as a deviation (in cents) from 12TET. As this approach requires making adjustments to each note, tuning becomes a time consuming exercise that raises the potential for mistakes and often requires a preconceived notion of what frequencies to select, stifling the ability to make intuitive decisions about the tunings.

An alternative approach defines tunings as a ratio to the first scale degree as shown in table 1. Whilst this method lacks the anchor point of 12TET deviation, it gives a far more elegant representation of the scale. Just intonations in particular are represented clearly in this way due to their inherent use of natural numbers for ratios.

2.1 Equal Temperament

Equal temperament creates a perceptually consistent width interval [1] between each note. This creates a harmonic compromise with partials from each note close enough to integer multiples to be perceived as harmonic, without creating larger gaps that are perceived as inharmonic[2] at other points in the chromatic scale.

This tempering system works by dividing an interval range (typically an octave) into equal divisions. In western traditional theory this is typically twelve subdivisions following the formula:

$$\sqrt[12]{2} = 2^{(1/12)} \approx 1.05946309436$$

It is possible to divide the octave into more than twelve notes and this is the most typical choice for exploring microtonal music in the 21st century. Again temperament, by its compromise facilitates the use of the full scale and therefore key changes, for physical instruments. The concept of tempering a scale is to provide this functionality. Interestingly, Hinrichsen has suggested that equal temperaments would benefit from a wider interval width, that would provide a more harmonic series (though sacrificing the tuning of the octave itself) [5].

2.2 Just Intonation

Just intonation refers to tunings where the fundamental frequency of each note is related as an integer ratio to some common reference pitch. This concept creates a set of harmonics that align in an audibly consonant way and is by strict definition, what is considered *'in tune'*.

As this series of notes is built on integer ratios, this relationship only holds for the current key[3].

Just intonation is more accurately conceptualised as a collection of relationships rather than a single rigid tuning. There are variations such as *Pythagorean Tuning* (table 1) and *Five-limit tuning*, handling different scale degrees with different ratios, creating subtly different qualities.

3 CURRENT INTERACTIONS WITH INSTRUMENT TUNING

Currently there are a number of synths that support the ability to create alternate tunings. Whilst there is support for altered tunings within the MIDI standard (via SysEx message), it is the accessibility and quality of digital instruments that present the best way to overcome the physical tuning limitations of traditional instruments.

The major limitation in tuning arises from the limited ways to derive, experiment and configure tunings rapidly enough to fulfil the creative needs.

Tuning by deviation from 12TET has been used by the likes of Terry Riley in "Songs For The Ten Voices Of The Two Prophets" [10] to explore tuning in a more contemporary context, using Prophet 5 Synthesisers. Riley's work stands as a fairly experimental piece that goes to very deliberate efforts to explore the harmony not typically associated with keyboard instruments. It is a great example of what can be achieved but generally the process is demanding on the composers comprehension and understanding of theory, beyond that of western classical music theory or an intuition for harmonic relationships and exploration.

Whilst more recently work has been done on modelling microtonal tunings for 3D printed flutes[1], there appears little work that allows for more fundamental tuning concepts for both description of the tuning and the application to instruments.

Hayward [4] does present an interface to describe just intonations as a lattice. This demonstrates the use in visualising the abstract relationships in just tunings and thereby demonstrates an opportunity to creatively explore them. His paper does however focus exclusively on just intonations and this presents as more of an analysis tool than a tool which can be used directly for practical applications.

There is also work that explores the opportunity for dynamic tuning. For example, Milne presents an isomorphic controller that facilitates tuning adjustments during performance[9]. Milne's work certainly allows for tonal exploration however the price for this power is a technically challenging controller.

4 EXPRESSING TEMPERAMENT

Many functional languages have traits that are conductive to handling lists. Haskell, thanks to higher order functions, has the ability to perform a number of generalised actions over lists such as; mapping, zipping and filtering. These can allow a programmer to quickly but also arbitrarily describe and interact with lists representing tunings. This is highly beneficial as results can be computed

[1] Equal to 100 cents

[2] Referred to as wolf note - the limiting factor in key modulation when using just tunings.

[3] Even then, more than 12 notes are required, meaning enharmonic notes that are not equivalent.

Note	G♭	D♭	A♭	E♭	B♭	F	C	G	D	A	E	B	F#
Ratio	1024:729	256:243	128:81	32:27	16:9	4:3	1:1	3:2	9:8	27:16	81:64	243:128	729:512
Cents	588	90	792	294	996	498	0	702	204	906	408	1110	612

Table 1: Pythagorean tuning

and applied far faster than manually tuning per note, helping encourage the iterative workflow that is conductive for creative and intuitive exploration.

Given the fact that there are a number of ways to describe a tuning, a generic tool such as a programming language actually offers the designer the opportunity to approach tuning an instrument from a number of perspectives. Whereas for many instruments, tuning is bound as a variation from 12TET, using Haskell, lists can be generated and applied, through a series of expressions or application of a set of helper functions, as proposed later.

4.1 Formula Expression

For generating equal temperaments, creating an expression that divides the desired interval is a simple and effective method for creating a tuning. Equal temperament takes the interval of an octave and splits into twelve perceptually equal parts. This takes the form:

$$r = \sqrt[n]{p}$$

Where r is the ratio, p is the interval and n is the number of divisions.

This relationship can be used to create an expression that gives the frequency of a note given a scale degree d and a reference pitch R:

$$2^{(d/n)}R$$

To this extent, exploring equal tempered scales should be trivially simple as variations of this formula. Typical microtonal music can then be explored given any expression where $n > 12$.

4.2 Ratios and Lists

It is well recognised that ratios are a powerful tool for describing tunings, a good example being Wright's book on the mathematics of music [12]. Whilst for equal temperament an expression is an effective and simple way of expressing a scale, due to the variation in interval size, this is not so simple to work with for just intonations.

An alternative to formulaic expression is to describe a scale by its intervallic spelling, that is a list where each interval that constructs the scale is explicitly described as a ratio.

$$A = 1024/729, 256/243, 128/243 \ldots 242/128, 729/512$$

If this list[4] is evaluated what remains is a set of coefficients that can be multiplied with a reference frequency R to calculate the tuning.

$$f = RA_n$$

[4]Pythagorean temperament described as a list.

This presents a very intuitive way of experimenting with very strong harmonic tunings in a way that suits the human inclination for seeing patterns.

4.3 12TET deviation

Whilst we consider deviation from 12TET to be a limiting method for tuning, as this paper's approach is general, it is easy to include it, for both completeness and familiarity. Below n is equal to the number of cents up from the reference pitch.

$$f = R(2^1/1200)^n$$

5 PRACTICAL EXAMPLES

Given its light, mathematical syntax, these formula translate conveniently into functional languages such as Haskell. We present several very simple implementations here based upon Haskell's list comprehensions, where given a list of the desired scale degrees, a list of frequencies are returned. A list comprehension is a powerful programming construct, found in languages such as NPL, Miranda, and Haskell, that enables the concise and expressive construction of lists[11]. Using a list comprehension we can replicate the formula for generating an equal tempered scale of twelve chromatic notes with the expression below. The named constants make it a simple change in order to recreate the scale from a different reference point, for example, to tune to the popular alternative $A = 432Hz$.

```
referencePitch = 432
freq_12tet =
  [2.0 ** (n/12.0) * referencePitch | n <- [0 .. 13]]
```

Below we generate a Pythagorean circle of fifths using a list comprehension.

```
freq_PT = [referencePitch * (3/2) ** n | n <- [0..12]]
```

To create a scale using intervallic expression we simply create the list. This is then evaluated to be used in the list comprehension. The resulting coefficients can then be used in a list comprehension.This comprehension could actually be generalised as a function to apply scale coefficients. Further, an auxiliary function is also presented here that can be mapped over a scale to shift the scale up by an octave.

```
octatveUp x = x * 2
ratios_just = [1, 25/24, 9/8, 6/5, 5/4, 4/3, 45/32,
               3/2, 8/5, 5/3, 9/5, 15/8, 2]
freq_just =
  [ referencePitch * (ratios_just !! (n - 1))
                              | n <- [1..12]]
freq_just2 = map octaveUp freq_just
```

6 APPLICATIONS

6.1 Scale generation

Based on the presented examples, it is possible to interface with a number of existing Haskell harmony libraries and frameworks (such as Haskore [7]), taking the lists generated and using higher order functions such as filter to select or remove notes according to diatonic patterns found in those libraries. Alternatively the lists found in these libraries could also be used in the list comprehension itself to create predicate conditions or to select only values from the list of diatonic notes, such that scale notes are the only frequencies created.

6.2 Digital Instrument Tuning

Applying these concepts are intended as part of an ongoing work however it would be simple to apply the output of the implementation proposed here to an OSC system or even for use with MIDI. For example, the output could be formatted into a SysEx message that can be sent to configure a MIDI device that supports the MIDI standard's tuning.

7 CONCLUSION

It's notable to consider how little research has been carried out in the area surrounding intonation and temperament this century given that it provides the foundation upon which musical theory sits. This is particularly true more recently, when considering digital musical instruments and how they support tuning. While there are occasional examples touching on peripheral concepts, substantial literature around the ideas of tuning and tempering were far more popular through the 80's and 90's. This appeared to trail off in popularity and seems to be rarely considered at a fundamental level when designing new, expressive instruments.

Given the basic premises outlined here, it is hoped that these ideas can be applied when considering the design of new digital musical instruments, such that there is less limitation in the exploration of tonality. Exposing this level of functionality is highly beneficial given the number of people engaging with digital instruments and computers, especially as they are often less constrained by notions of western classical theory. Composing and creating on a intuitive level need not be bounded by the conventional theory that has underpinned, but perhaps constrained 100 years of music.

8 ACKNOWLEDGEMENTS

The authors would like to thank the reviewers for their constructive feedback and also the members of Computer Science Research Centre and Creative Technology Lab for many interesting discussions. In particular, Sam Hunt for providing useful comments on the final draft of this submission.

REFERENCES

[1] Matthew Dabin, Terumi Narushima, Stephen Beirne, Christian Ritz, and Kraig Grady. 2016. 3D Modelling and Printing of Microtonal Flutes. In *Proceedings of the International Conference on New Interfaces for Musical Expression (2220-4806)*, Vol. 16. Queensland Conservatorium Griffith University, Brisbane, Australia, 286–290. http://www.nime.org/proceedings/2016/nime2016_paper0056.pdf

[2] R.W. Duffin. 2008. *How Equal Temperament Ruined Harmony (and Why You Should Care).* W. W. Norton. https://books.google.co.uk/books?id=i5LC7Csnw7UC

[3] Alexander John Ellis. 1885. On the musical scales of various nations. *From The journal of the society of arts* 33, 1688 (1885), [485]–527 p.

[4] Robin Hayward. 2015. The Hayward Tuning Vine: an interface for Just Intonation. In *Proceedings of the International Conference on New Interfaces for Musical Expression*, Edgar Berdahl and Jesse Allison (Eds.). Louisiana State University, Baton Rouge, Louisiana, USA, 209–214. http://www.nime.org/proceedings/2015/nime2015_146.pdf

[5] Haye Hinrichsen. 2016. Revising the musical equal temperament. *Revista Brasileira de Ensino de FÃsica* 38 (00 2016). http://www.scielo.br/scielo.php?script=sci_arttext&pid=S1806-11172016000100410&nrm=iso

[6] Paul Hudak. 1996. Building Domain-specific Embedded Languages. *ACM Comput. Surv.* 28, 4es, Article 196 (Dec. 1996). https://doi.org/10.1145/242224.242477

[7] Paul Hudak, Tom Makucevich, Syam Gadde, and Bo Whong. 1996. Haskore music notation âĂŞ An algebra of music âĂŞ. *Journal of Functional Programming* 6, 3 (1996), 465âĂŞ484. https://doi.org/10.1017/S0956796800001805

[8] Simon Marlow. 2010. Haskell 2010 Language Report. (2010).

[9] Andrew Milne, William Sethares, and James Plamondon. 2007. Isomorphic Controllers and Dynamic Tuning: Invariant Fingering over a Tuning Continuum. *Computer Music Journal* 31, 4 (2007), 15–32.

[10] Terry Riley. 1983. Songs For The Ten Voices Of The Two Prophets. (1983). Format:CD.

[11] D Turner. 1986. An Overview of Miranda. *SIGPLAN Not.* 21, 12 (Dec. 1986), 158–166. https://doi.org/10.1145/15042.15053

[12] D. Wright. 2009. *Mathematics and Music.* American Mathematical Society, Chapter 4, 45–46. https://books.google.co.uk/books?id=g4ONAwAAQBAJ