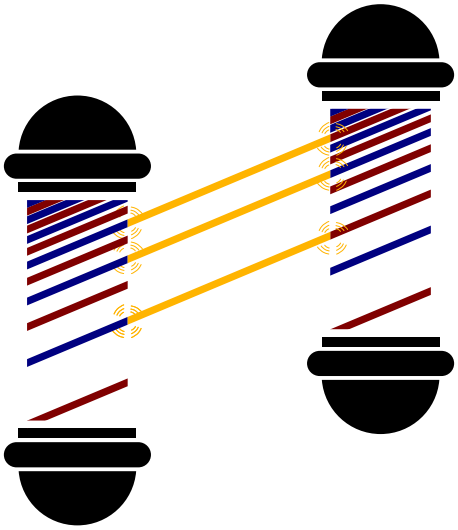


# An Adaptive Just Intonation Algorithm for Barbershop Music

**Teun Buwalda**
t.c.buwalda@students.uu.nl
6787886, Utrecht University
Bachelor's thesis

April 22, 2023

**Abstract**

When harmonies in music are tuned to follow just intonation intervals, it is impossible to always keep held notes at the same pitch while also maintaining a steady tonal centre. Adaptive tuning systems make specific choices to optimise those constraints, but none have been made to specifically work for barbershop music. The first part of this thesis distils an algorithm that optimises for the tuning conventions of barbershop. The second part implements this algorithm in C♯ as a program that takes a manually tagged barbershop score as input and outputs a MIDI file with pitch bend messages. The proposed system provides just intonation at each moment in time and furthermore optimises three parameterised constraints: close-to-equal-temperament intervals in the lead voice, minimal retuning of held notes and minimal tonal centre drift. Results show that the algorithm performs well on example songs and the provided parameters have a significant effect on the resulting tuning. All code is publicly available on `GitHub`.

***Acknowledgements***

*Thank you to Jeroen Fokker and Peter van Kranenburg for their guidance throughout the project; to Jos Mulkens for his continuous support and wise thoughts; to Tijs Krammer for setting me on the course of an adaptive tuning system; to Jasper de Gier and Jay Dougherty for their expertise from the barbershop world; to Than van Nispen and Marc Groenewegen for their artistic input; to Titus Mars and Jeanne Buwalda for their critical view.*

# 1 Introduction

## 1.1 Adaptive Just Intonation for Barbershop

The piano is out of tune. In many musical contexts, *just intonation* is seen as the ideal tuning method, in which distances between tones follow simple mathematical frequency ratios. [1], [2] At the same time, western music theory recognises twelve tones per octave. [3] Sadly, tuning those twelve tones to fixed pitches such that they all follow just intonation is mathematically impossible. (2.1) The modern piano, and most modern music with it, therefore use a system called *equal temperament*, which only vaguely approximates the frequency ratios from just intonation. [4]

Music groups that have the possibility to, can attempt to alter their intonation based on the playing chord, to still achieve just intonation. *Barbershop* (2.3) is a genre of four-part music primarily sung in quartets, who explicitly try to achieve just intonation on every chord. [5], [6] *Adaptive tuning systems* are algorithms that automatically tune each playing note such that they approximate just intonation. [7] However, when a singer holds a note across multiple chords, there is no unbiased answer as to how exactly each note should be tuned. Holding the note at exactly the same pitch can either require a deviation from just intonation or an overall pitch drifting effect over time (2.2).

There has been extensive research into adaptive tuning systems. [8] Most of this previous work takes a general approach, maximising dynamicity to work for live music, but they are not quite applicable to barbershop. For example, Hermode Tuning [9] and Pivotuner [10] are commercially available plug-ins that can retune piano music as it is being played. As a consequence, adaptive tuning systems generally do not model human choirs. In particular, an algorithm that takes the limitations of barbershop quartets into account, such as the roles of the four different voices, is not known to have been made to the author.

This thesis project will answer two separate questions. Question A concerns the solution to barbershop quartets' problem: *what adaptive tuning algorithm would a mathematically ideal barbershop quartet follow, given a score?* In this first part of the thesis, an adaptive tuning system for barbershop music will be proposed. It will optimise four criteria (outlined in Section 3) which are important to the genre. In order to validate the algorithm, the Results section (6) will show how the algorithm accounts for these four criteria and where concessions need to be made in order to minimise conflicts between them.

Research question B is: *can we implement the tuning system from question A so that it models the ideal barbershop quartet?* This second part of the project should result in a C♯ program that, given a sequence of chords, can play them back justly tuned and following the genre's limitations, possibly serving as a suggestion for performers and allowing for high-level analysis of the algorithm's performance in Section 6.

To provide the necessary context to answer both of these questions, the following section will expand upon the subjects of just intonation, adaptive just intonation algorithms and barbershop music. The following section will describe the exact methodology and further describe the algorithm's constraints. Sections 4 and 5 will answer research questions A and B, respectively. Section 6 will evaluate the algorithm by analysing results such as global pitch drift and particularly hard-to-tune passages. Section 7 will then place this research in the context of other literature about choirs, tuning and philosophy.

## 1.2 Relevance to KI

This thesis project is part of the bachelor Kunstmatige Intelligentie at Utrecht University. The Utrecht University AI programmes focus on Human-Centered Artificial Intelligence (HCAI), rather than trying to keep up with the latest innovations in machine learning algorithms. HCAI targets to understand, reproduce and if possible enhance human intelligence. [11] Modelling something specifically human such as music is inherently AI research, especially when trying to build an algorithm that is inspired by human performance, but performs better. Therefore, finding a tuning algorithm for barbershop music fits

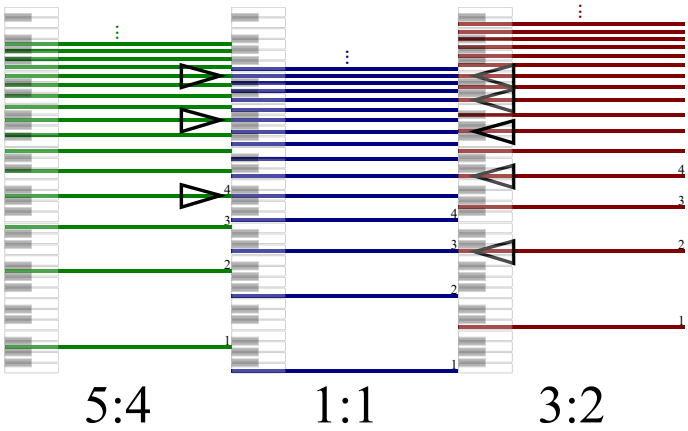



**Figure 1**: The relationship between just intonation and the overtone series, which is the cause of the "lock and ring" effect as pursued by barbershop musicians. The figure shows three tones with their respective overtones, the harmonic frequencies that create the sound's timbre. A piano keyboard was overlaid for pitch reference. The A (left, green) is tuned a just major third above the F (middle, blue), which makes the A's 4th, 8th and 12th overtone share the exact frequency of the F's 5th, 10th and 15th. The C (right, red) is tuned to a just fifth above the F, which means every one of its overtones which is a multiple of 2 shares its frequency with one of the F's overtones which is a multiple of 3. Because of these shared overtones, the chord has a stable sound, minimising interference by close-but-not-quite-the-same frequencies.

right into the HCAI focus area.

This study involves the definition of a new language to describe written music, was written as a computer science project and analysed afterwards from a philosophical perspective. All three of those sciences are important areas of HCAI.

# 2 Theoretical Framework

## 2.1 Just Intonation

In many musical contexts, *just intonation* is seen as the ideal method for tuning. [1] Rather than predefining each note's pitch, like necessary when tuning an instrument such as the piano, just intonation defines specific frequency ratios for each musical interval. The most common intervals are the octave (defined as $2:1$), the fifth ($3:2$) and the major third ($5:4$). For example, if an A is tuned at 220 Hz, then an E a fifth above this A will be tuned at $220 \cdot \frac{3}{2} = 330$ Hz. The minor seventh, which is also prevalent in barbershop music [5], is defined as $7:4$. [4] These ratios are different from the tuning system used by most modern pianos.

The ratios are drawn from the simplest intervals of the *harmonic series*, which refers to the set of overtones that can be heard above nearly every sung, played or synthesised tone. When musical intervals follow these simple ratios (i.e. with a small nominator and denominator), the overtones of different tones "lock together" and create a specific auditory experience that many musicians strive for. This sound is caused by the fact that tones which share a simple ratio will also share many overtones (Figure 1). Because of these shared overtones, the different tones barely interfere with each other, creating a more stable sound.

Sadly, it is mathematically impossible to devise fixed pitches for a set of twelve tones such that each pair of tones always exactly follows one of the simple ratios from just intonation. A consequence of this fact is that it is also impossible to provide a "perfect" tuning for the piano. This impossibility is

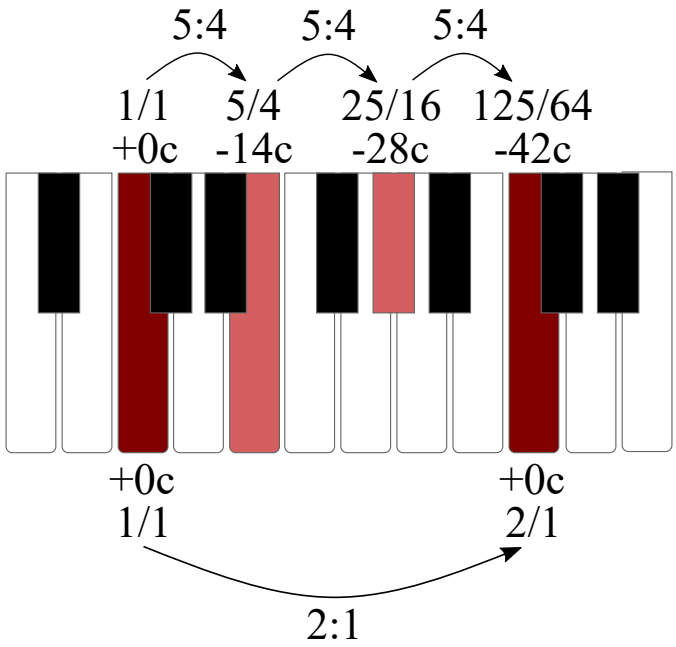



**Figure 2**: 12 fixed tones are not compatible with just intonation. For each coloured tone, its frequency ratio with the low C is written. The difference with equal temperament in cents is also written. The bottom path shows that a high C should be tuned at exactly twice the frequency of the low C when tuning the octave justly. The top path instead uses justly tuned major thirds with the standard $5:4$ ratio. The two paths come out at a different tuning for the high C, proving that major thirds and octaves are incompatible.

demonstrated with the example in Figure 2, which shows two different ways to walk from a low C to a high C. Suppose we want the major third to be equal to 4 semitones, as is the case in western music theory [12], then a major third should sound between C and E, between E and G♯ and between G♯ and a high C. The major third has a ratio of $5:4$. Then the ratio between a low C and E on the piano should be $5:4$, as should the ratio between E and G♯ and the ratio between G♯ and a high C. Therefore, the ratio between the low C and G♯ should be $\frac{5}{4} \cdot \frac{5}{4} = \frac{25}{16}$ and the ratio between the low C and the high C should be $\frac{25}{16} \cdot \frac{5}{4} = \frac{125}{64}$. However, the two Cs are also one octave apart and should therefore have a ratio of $\frac{2}{1}$. These two ratios are close, but not exactly the same, which shows that it is impossible to devise a tuning system such that both major thirds and octaves are always in tune according to just intonation.

A possible solution to the fact that just intonation is impossible to achieve with twelve fixed pitches, is to *temper*, or approximate, the intervals of the harmonic series. Most modern music has accepted *twelve-tone equal temperament* as the best substitute for just intonation. In twelve-tone equal temperament, the octave is divided into twelve logarithmically equal divisions. Since the octave is defined by the frequency ratio $2:1$, this custom puts the smallest interval (the semitone) at exactly $\sqrt[12]{2}:1$. [4] For example, if an A is tuned at 220 Hz, then the lowest tone above this A, a B♭, will be tuned at $220 \cdot \frac{\sqrt[12]{2}}{1} \approx 233.08$ Hz. Figure 3 shows the standard tuning in equal temperament for a part of the piano keyboard. Twelve-tone equal temperament works relatively well for some intervals, such as the fifth, which is only slightly too small compared to just intonation. On the contrary, intervals like the major third and the minor seventh are significantly smaller in just intonation than in equal temperament.

When talking about the difference between just intonation and 12-tone equal temperament, the *cent* interval (c) is often used as a unit. 1 cent is defined as $\frac{1}{100}$ of a semitone in equal temperament, meaning it has a frequency ratio of $\sqrt[1200]{2}:1$. A justly tuned major third, for example, is 14 cents lower than a major

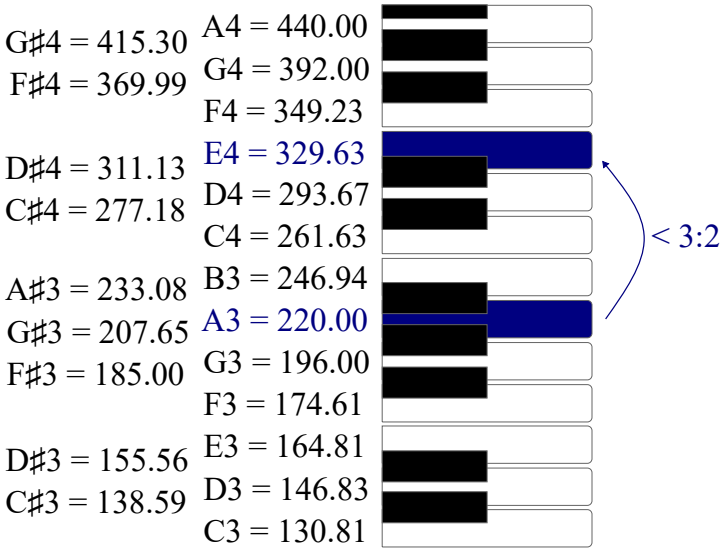



**Figure 3**: A selection of standard frequency values for piano in equal temperament, shown in Hertz. Equal temperament is the most common tuning method and ensures that each semitone is always the same relative size, namely $\sqrt[12]{2} : 1$. However, just intonation is not compatible with equal temperament; for example, if this E4 were tuned to a perfect fifth $(3 : 2)$ with the A3 according to just intonation, it should be 330.00 Hz, which is slightly higher than it is in equal temperament.

third in equal temperament (Figure 2).

*Note about terminology.* The addition of the cent interval leaves us with three different ways to describe a tone's exact tuning: as an absolute frequency in Hertz, as a fraction relative to a base note or as a logarithmic difference with a base note in cents. Exact frequencies are generally useful in phsics: an A3 can be tuned at 220 Hz, whereas an E4 would be tuned at 329.63 Hz on the piano. To describe just intonation, we often use fractions: when tuning an E4 to be a perfect fifth above this A3, the E4's tuning would need to be $\frac{3}{2}$ times the frequency of the A3 (equivalently, 330.00 Hz). The cent interval can put exact tuning in a musical perspective by comparing to equal temperament: 330.00 Hz is 2 cents higher than 329.63 Hz, meaning $\frac{2}{100}$ of a semitone.

## 2.2 Adaptive Just Intonation Systems

Although just intonation is impossible to achieve when all twelve piano keys have a fixed tuning, such as on an acoustic piano, it can be achieved when using instruments that can *adaptively* tune specific notes. In other words, if a note can be tuned differently based on which other notes are playing, the strict mathematical ratios of just intonation can be followed. This idea is, for example, applied to ensembles with human voices, string instruments such as the violin, or trombones, since all of those instruments can continuously change their pitch by ear. [4] Modern computational techniques also allow digital instruments to be tuned adaptively. [8]

However, there is no single algorithm that completely solves the mathematical problem of just intonation. Choices need to be made with regards to optimisation of melodic intervals, held notes and tonal centre drift. Optimisation of melodic intervals means that the intervals within the melody follow equal temperament as much as possible, making melodies sound more intuitive. [13] Optimisation of held notes means notes that keep sounding in different chords do not need to be dramatically retuned for each new chord. Tonal centre drift is a gradual lowering or heightening of the average tuning of notes, which

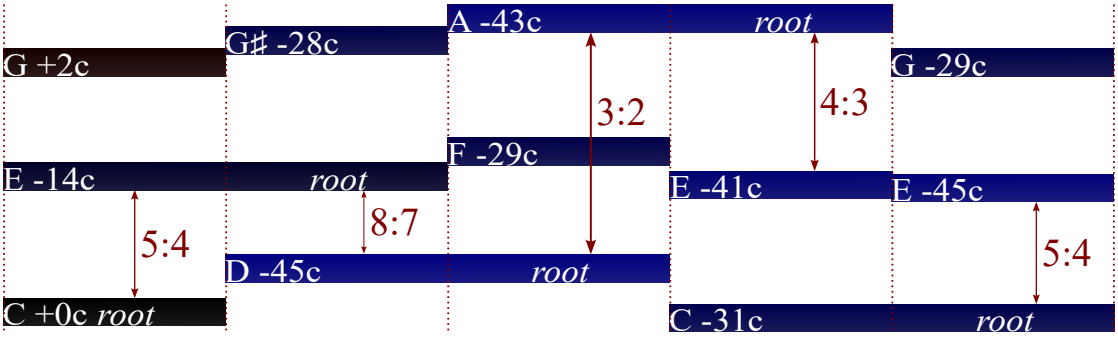



**Figure 4**: Incompatibility between the just intonation, tie and drift constraints. Each note shows its note name and deviation from equal temperament in cents. The just intonation fraction is also shown for some intervals. The held notes in combination with just intonation intervals require the overall tonal centre to shift downwards.

is generally seen as undesirable. [5] If held notes are allowed to retune freely, just intonation without pitch drift is possible. To achieve this goal, the performance can make sure the melody always follows equal temperament intervals and the other voices change their pitches to tune justly with the melody. [13] On the contrary, the constraint of held notes staying at the same pitch between different chords causes problems in tonal centre drift.

To illustrate this incompatibility of the held notes constraint with the other constraint, consider the scenario in Figure 4. In this scenario, some notes are held over changing chords. The initial C-E third requires the E to be tuned 14 cents lower in order to create the just $5:4$ ratio. Because this E is held into the next chord, the other notes need to be tuned even lower. Eventually, these held notes cause the last C to be tuned 31 cents lower than the initial one. Also, because of the different functions of the last two Es, an audible pitch shift needs to be made between them. The overall tonal centre has moved 31 cents downwards, (*flat*). This downward shift shows that held notes and tonal centre drift cannot both be optimised at the same time. As another example, the music in Figure 6 is an adapted version of a famously "rising" piece of music, developed by music theorist Benedetti (1530-1590). [14]

Much research has been done into automatic *adaptive tuning systems*: computer programs that take sets of musical notes as input, and return their respective tunings according to just intonation as output. [8] Sethares [7] approached the problem very generally by minimising a loss function based on a sound's dissonance curve. Løberg [15] implemented a dynamic tuning system by the Norwegian composer Eivind Groven that dynamically chooses between 36 available tones to get just intervals. Schwär et al [16] developed a loss function for intonation in order to define it as a cost minimisation problem. Horner and Ayers [17] developed a genetic algorithm for tuning that also optimises for held notes. Commercial software such as Hermode Tuning [9] and Pivotuner [10] can be integrated into music software to retune all played notes based on the sounding chord. Hermode sums the deviations in cents from equal temperament of all sounding notes and ensures that the result is mostly equal to 0. Conversely, Pivotuner picks a single note in each chord that should remain at the same pitch, enabling an artistic use of the resulting tonal centre drift.

Most of the above-mentioned just intonation algorithms follow a general strategy for tuning, to apply to all genres. An algorithm that takes the limitations of barbershop quartets into account, such as the roles of the four different voices, is not known to have been made to the author.

## 2.3 Barbershop Music

Barbershop is a distinct genre of four-part singing that is most prevalent in North America that is primarily sung in quartets. [18] The Barbershop style is very specifically defined and has an active community

that aims to preserve the genre according to its general definition. Some of the largest central events where Barbershop enthusiasts gather are competitions, where quartets are judged according to this specific definition. In this thesis, the Barbershop Harmony Society's definition of barbershop music and its Contest and Judging Handbook [5] will often be cited as the main source of information about priorities within the genre.

In this official definition, barbershop music is described as having four voices: tenor, lead, baritone and bass. The lead generally sings the melody, but exceptions can occur within a song. The melody is accompanied by mostly homorhythmic (i.e. in the same rhythm as the lead) harmonies in the other three voices. Chords generally do not get too complicated: most songs revolve around major and minor chords without many harmonic extensions. [5] Additionally, the dominant seventh chord is often described as the most important chord in barbershop music. [6]

The Contest and Judging Handbook (page 7-2, paragraph II.A.) describes just intonation as one of the key elements of barbershop music. It prescribes tuning in barbershop music as follows:

> Barbershop singers adjust pitches to achieve perfectly tuned chords, and yet sing a melodic line that remains true to the tonal center. Barbershop singers strive for more precise tuning than is possible with the fixed 12-tones- per-octave of the equally tempered scale of fixed-pitched instruments, such as the piano. Essentially, we use just intonation for harmonic tuning while remaining true to the established tonal center. [5]

Barbershop singers try to minimise "beats" in the sound of their harmonies, the auditory artefacts that appear when chords are not tuned justly. When quartets follow interval ratios as described in 2.1, the overtones of the different parts match and create a buzzing, unchanging auditory experience that is described as "lock and ring".

However, very few barbershop quartets consistently achieve the high standard of just intonation. [18] The above quote immediately describes one of the major dilemmas that singers face when attempting to follow just intonation: tonal centre drift (see 2.2). Besides attempting to maintain a constant tonal centre, quartets may also try to have the lead follow consistent melodic intervals similar to those on the piano, and refrain from moving held notes around too much.

# 3 Methodology

This thesis will propose an algorithm that adaptively tunes notes in barbershop scores to just intonation. Figure 5 shows the pipeline that the implementation should follow. The tuning system will optimise the following criteria:

1. maintain just intonation on every chord [*just intonation constraint*]
2. account for the different roles of the four parts: specifically, the lead's intervals should mostly stick to 12-tone equal temperament [*lead constraint*]
3. minimise (but not necessarily eliminate) retuning of held notes [*tie constraint*]
4. minimise (but not necessarily eliminate) tonal centre drift [*drift constraint*]

The input of the algorithm is basically the score of a barbershop song. Additionally, the music is tagged with information about the intended harmonies, as those can be relevant for tuning. The output is a tuning for every note that optimises the above four constraints.

For the C♯ implementation, the input already will already need to be split into a format that is friendly to the algorithm's internal representation. This representation will be different from existing digital representations of music [19]–[21], because it will already contain information that is specifically applied to

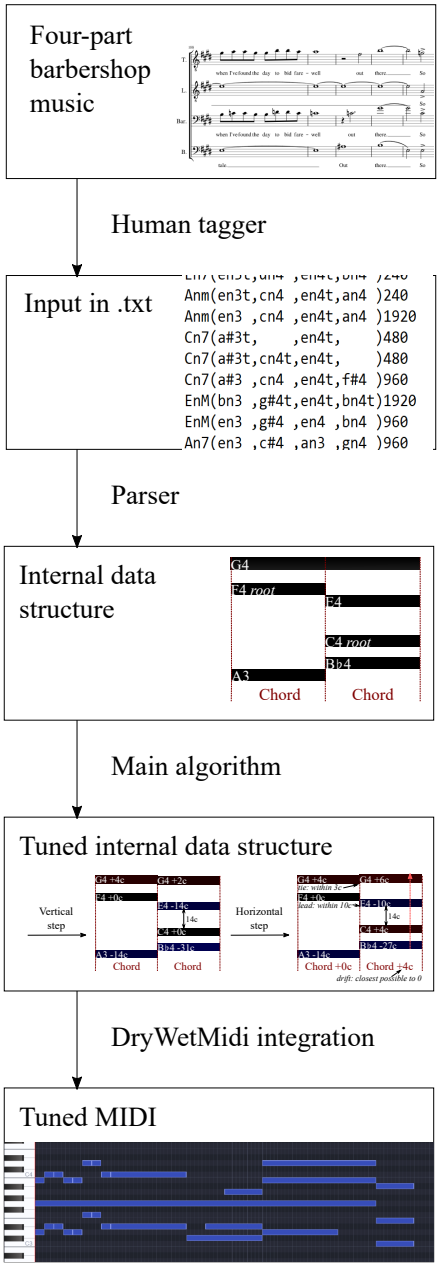



**Figure 5**: Pipeline for the implementation of the proposed adaptive just intonation algorithm for barbershop music. The program works on any four-part score, but requires a human tagger to translate it into a language that is specific to this algorithm. A parser can then interpret this language and build the internal data structure that the main algorithm uses to determine how each note should be tuned. At last, the library DryWetMidi is used to convert the tuned internal data structure to a MIDI file with pitch bend messages.

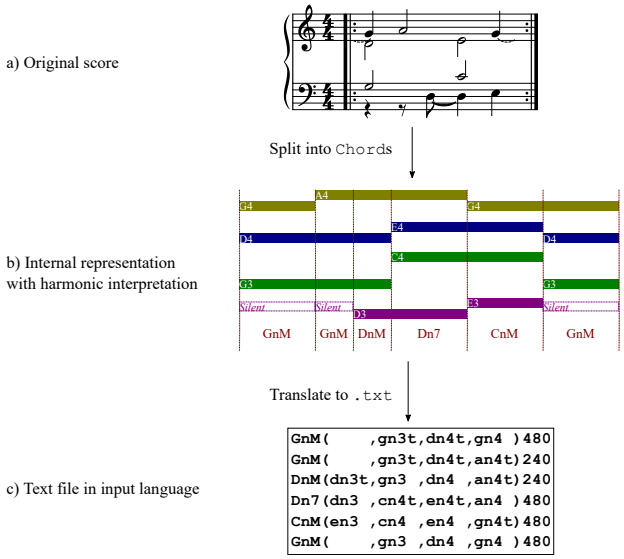



**Figure 6**: Example of a song in the input language. Human involvement is necessary to convert the music to the algorithm's internal data representation, especially to add chord symbols for intended harmonies.

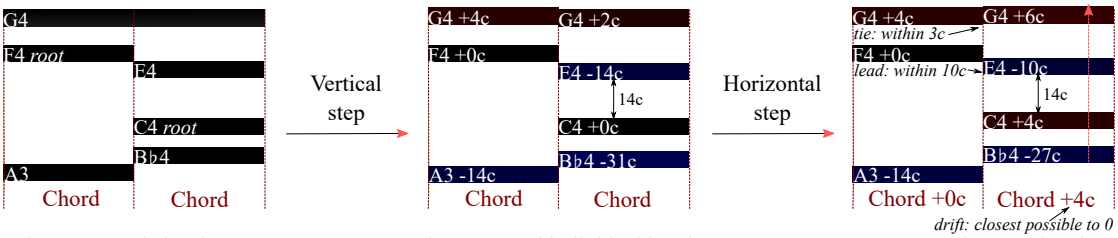
this tuning algorithm. The input should therefore be written in `.txt`-files using a newly-defined language. This language is described in Appendix 6. Importantly, the input splits the score into `Chords`, which in this case are periods of time where none of the notes change. A new `Chord` begins any time one of the four voices stops singing, starts singing or moves to a different pitch. Figure 6 visualises such a split score. Because of this newly-defined language, a parser must be programmed to convert the input `.txt` files into the program's internal data structure. Conversion from the original sheet music to the input language is, for now, a human job.

The output of the C♯ consists of a MIDI file [19] and some statistics, such as how much the song had to drift in general and which chords caused the most dramatic retunings. The MIDI files mostly contain the original notes from the score, now preceded by pitch bend messages. Section 5 will describe the MIDI conversion in further detail.

# 4 Algorithm

This section will describe the tuning algorithm proposed by this thesis, thereby answering research question A: *what adaptive tuning algorithm would a mathematically ideal barbershop quartet follow, given a score?* The algorithm annotates each note in the song with a bend value that prescribes how much it should deviate from 12-tone equal temperament. The algorithm consists of a vertical step and a horizontal step. The vertical step satisfies the just intonation constraint from Section 3, whereas the horizontal step satisfies the other three constraints. The vertical step is a relatively standard procedure for just intonation, whereas the horizontal step is a completely new proposal. Figure 7 shows an outline of the full algorithm.

The algorithm distinguishes between two levels of tuning: *individual bend* and *master bend*. The individual bend describes how much the notes within a chord differ from each other relatively and is determined in the vertical step of the algorithm. A note's individual bend is only relative to the tuning of the other three notes in the chord, where a chord only spans a period of time where not a single note changes. The master bend, on the other hand, is used to move an entire chord's tuning up or down and is determined in the horizontal step. A note's pitch is ultimately calculated by adding its individual




a) input: tagged chords b) computed individual bend c) computed master bend

**Figure 7**: A basic view of the algorithm. tieRange = 3c, leadRange = 10c. The vertical step tunes each note separately, relatively to each other (*individual bend*), satisfying the just intonation constraint. The horizontal step then attempts to satisfy the tie, lead and drift constraints by moving all notes in the chord at the same time (*master bend*).

bend to the chord's master bend (*posterior bend*) and using the result as a degree of deviation from equal temperament. Importantly, a note's individual bend never changes after the vertical step of the algorithm is done, making sure the notes of one chord always stay justly tuned with one another. Therefore, constraint 1 is always satisfied.

The vertical step (Figure 7b) determines each note's individual bend using just intonation intervals. The bend value for the chord root is always set to 0 (again, since these are relative values, that does not actually mean the root will be tuned to the regular 12-tone equal temperament value). The bend values for the other playing notes are decided using a set of interval values based on fractions relative to the chord root (Figure 8). This set of fractions can be given by the user, but a standard set is provided. The fractions are taken from simple ratios in the harmonic series. For example, the dominant seventh chord is usually described with a $4:5:6:7$ ratio between its chord tones. Therefore, in an A dominant seventh chord (A-C♯-E-G), the G's frequency will be tuned to be $\frac{7}{4}$ times the A's frequency. On the other hand, a half-diminished chord is usually described with a $5:6:7:9$ ratio, so in an A half-diminished chord (A-C-E♭-G), the G will be equal to $\frac{9}{5}$ times the A. Differences like this one are why the chords need to be tagged with their intended harmony.

The horizontal step (Figure 7c) moves all chord notes around evenly to satisfy constraints 2-4 (lead, tie and drift; see Section 3). Specifically, it outputs a master bend value for the current chord by comparing it to the previous chord. For the lead and tie constraints, a collection of lead notes and tied notes from the previous chord is made in order to calculate how much the current master bend can move around while satisfying those constraints. Each of those notes has a desired space that the new note should move into. To illustrate: in the figure, *tieRange* is set to 3 and the previous G4 is tied over, so the current G4 should be within 3 cents of the old G4. Meanwhile, *leadRange* is set to 10 and the previous F4 has a posterior bend of 0 cents, so the current E4 should deviate less than 10 cents from equal temperament. After the collection of lead and tied notes has been completely run through, the algorithm chooses a value for the master bend that is as close to 0 as possible. This last step helps the drift constraint, because a master bend value of 0 means that there is no tonal centre drift.

Ultimately, the algorithm's results depend on a set of parameters given by the user. The first parameter is a set of lists of fractions that should be used for the vertical step (Figure 8). This first parameter is important to note, because opinions vary as to which exact tunings are the best for each interval. The other three parameters are referred to as *priority*, *tieRange* and *leadRange*. *priority* is either "lead" or "tie" and determines whether the lead or tie constraint should be satisfied first. *tieRange*, given in a unit similar to cents, determines how much a tied note is allowed to be retuned compared to its predecessor. *leadRange* is similar: it determines by how many cents an interval in the lead voice is allowed to deviate from a relative interval in equal temperament (i.e. from a multiple of 100 cents). For instance, if *tieRange* is set to 3 and *priority* is set to "tie", the horizontal step of the algorithm will prioritise moving the most important tied note less than 3 cents up or down over the other tied notes, the lead constraint and the drift

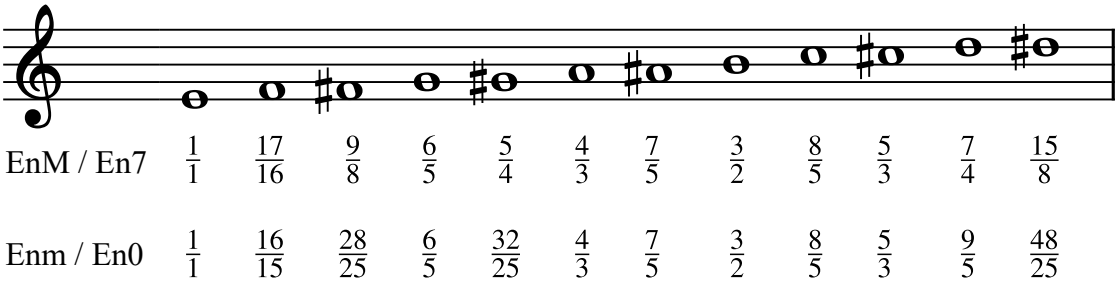



**Figure 8**: Different interval ratios for different chord symbols with the same root. The fractions are intervals in just intonation relative to the root note, the E. The intervals for major/dominant chords are chosen such that a dominant seventh chord follows the preferred ratio of $4 : 5 : 6 : 7$. [4] The fractions for the minor/half-diminished chords, on the other hand, are chosen such that a half-diminished chord follows $5 : 6 : 7 : 9$ and a minor seventh chord follows $10 : 12 : 15 : 18$. [4] These fractions are the default ones used by the vertical step of the algorithm, but can be changed using the language described in Appendix B.

constraint.

# 5 C♯ Implementation

Now that Section 4 has described the just intonation algorithm for barbershop music, we can look at how the algorithm was implemented into the programming language C♯. This section therefore answers research question B: *can we implement the tuning system from question A so that it models the ideal barbershop quartet?* The result of this section is a C♯ program that takes a song as a `.txt` file written in the language defined in Appendix A as input and outputs a MIDI file which represents the same songs, now with added pitch bend information. All code can be found on this thesis' `GitHub repository`.

The implementation consists of five classes: `Note`, `Chord`, `Song`, `BSTuner` and `Program`. Additionally, two new structs are essential to the algorithm's workings: `Fraction` (vertical step) and `Range` (horizontal step). `Note`, `Chord` and `Song` together form the internal representation of a Barbershop song. A `Song` contains an array of `Chords`, which in turn contains an array of exactly four `Notes` - if a part is silent in a `Chord`, its `Note`'s `playing` property is simply set to `false`. `BSTuner` contains the core of the algorithm and operates on a `Song` to tune the notes it contains. `Program` contains the `Main` method and serves as an interface for the program.

The vertical step of the algorithm is basically carried out by `BSTuner.SetIndivBends` and `BSTuner.GetIndivBend`, which set the `indivBend` property for each `Note` in a `Chord`. This `indivBend` property never changes again, meaning the logarithmic distance between the chord notes remains the same and therefore the just intonation constraint is always satisfied. The methods for the vertical step use the `Fraction` struct, which represents a frequency ratio as described in Paragraph 2.1, relative to the chord root. The `Fractions` used by the algorithm depend on the user input of the chord and chord type; Appendix B describes how the desired intervals can be described. The exact frequency ratios to use can be determined by the user in `.txt` files, as described in Appendix B.

The horizontal step of the algorithm is basically carried out by `BSTuner.SetMasterBend`, which sets the `masterBend` property for a `Chord`. The input for this method includes the *tieRange*, *leadRange* and *priority* parameters. *tieRange* and *leadRange* represent how "out of tune" the new intervals can be while satisfying the tie and lead constraints, respectively. *priority* prescribes which of the two constraints should be satisfied first. To achieve this optimisation of constraints, `SetMasterBend` uses the `Range` struct, which represents a range of bend values (represented logarithmically, in equal temperament semitones). `Ranges` are used to find a `masterBend` that satisfies the lead, tie and drift constraints for this `Chord`.

```
1: procedure SETMASTERBEND(MB_{i-1}, Notes_{i-1}, Notes_i, tieRange, leadRange, priority)
2:     ties ← [all notes in Notes_{i-1} that have the tie property, ordered lead-bass-tenor-baritone]
3:     ranges ← [an empty list of ranges]
4:     in order of priority:
5:         ranges.Add[for all tied notes, the range MB_i could move into in order to satisfy tieRange for this note]
6:         ranges.Add(the range in which MB_i could move according to leadRange)
7:     if 0 is within ⋂ranges then
8:         return MB_i ← 0
9:     else
10:        currRange ← ranges[0]
11:        for all range r in ranges[1:] do
12:            overlap ← r ∩ currRange
13:            if overlap == ∅ then
14:                return MB_i ← the number closest to r within currRange
15:            else
16:                currRange ← overlap
17:        return MB_i ← the number closest to 0 within currRange
```

**Figure 9**: Pseudocode for the horizontal step. The input is two Chords, both of which contain four notes. The output is a master bend value for the current chord which takes the tie, lead and drift constraints into account. $MB_i$ refers to the master bend of chord $i$, relative to equal temperament. The just intonation constraint has already been completely satisfied by the vertical step. Because this method compromises between the current range and the next once no overlap can be found, the drift constraint (Section 3) may be ignored if other constraints are not compatible with each other.

Pseudocode for the horizontal step can be found in Figure 9. The master bend method intersects ranges with each other until no overlap can be found. If all ranges overlap, `masterBend` can be set to the closest number to 0, meaning the amount of tonal centre drift has been minimised while all other constraints are satisfied as well. If one note's range does not overlap with the current range, the function ends by returning a `masterBend` within the current range that is closest to the unsatisfiable range, ignoring the drift constraint.

After the entire `Song` has been tuned, `Song.MIDISong` and `Song.WriteMidiFile` create a MIDI file out of it using the DryWetMidi library [22]. Each voice gets its own channel, meaning pitch bend messages only apply to a single voice. Every single `Chord` gets four pitch bend messages. The bend value for these messages gets calculated by adding the `Note`'s `indivBend` to the `Chord`'s `masterBend`. Since MIDI bend values only go up to two semitones [19], the MIDI key for a note may be changed if its bend value exceeds 2 semitones.

# 6 Results

This section will show the proposed algorithm's effect on an example score, namely the tag of *Ring-a-Ding Ding*, arranged by Anthony Bartholomew. Several good recordings of this song can be found on `YouTube`. Figure 10 shows the sheet music for this tag. [23] Since both the bass and tenor voices hold notes for a long time while the lead and baritone sing phrases in-between, *Ring-a-Ding Ding* is a good example to show how the proposed algorithm makes choices in ambiguous scenarios. This thesis' `GitHub repository` also contains the algorithm's output MIDI files for *Ring-a-Ding Ding*, as well as the results from other barbershop snippets, so it is possible to listen and compare.

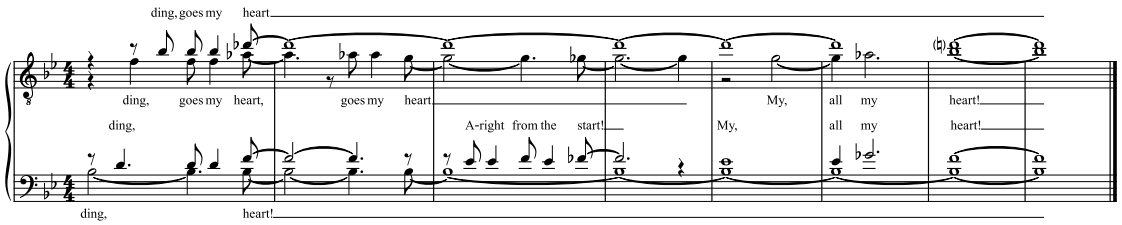



**Figure 10**: Sheet music for the tag of *Ring-a-Ding Ding*, a barbershop arrangement by Anthony Bartholomew.

Figure 11 shows the effects that the three parameters (tieRange, leadRange and priority, see 4) have on the lead, tie and drift constraints in *Ring-a-Ding Ding*. Note that the just intonation constraint is, because of the permanent nature of the vertical step, always satisfied. The lead constraint is quantified as the number of lead intervals that had to deviate more than 10 cents from equal temperament intervals. The tie constraint is quantified as the number of pairs of held notes where the second note was more than 3 cents higher or lower than the first. The overall tonal centre drift with these parameters has also been plotted.

The plots show that tonal centre drift is negatively correlated with both *Range* parameters: if notes and intervals are allowed to differ more from the ideal changes, then the song will drift less. Specifically, a stricter *tieRange* causes less pitch drift earlier than a stricter *leadRange* for this song. Since an "audible retuning" is set to be at least 3 cents, the number of audible retunings leaps after the *tieRange* passes 3 cents. The song only needs to deviate from equal temperament in lead intervals a couple of times, which is why this leap is invisible in the red line. When the *priority* is set to "lead", not a single deviation from equal temperament is necessary and the number of tied note retunings is relatively low in general.

# 7 Discussion

## 7.1 Results evaluation

Research question A was: *what adaptive tuning algorithm would a mathematically ideal barbershop quartet follow, given a score?* In this thesis project, an algorithm was proposed as the ideal tuning system for barbershop quartets, immediately approaching an answer to this question. The results show that there is no definitive answer to question A; it is impossible to tune the *Ring-a Ding Ding* tag in such a way that all four constraints from Section 3 are completely satisfied. However, the proposed algorithm successfully optimises the four constraints according to given priorities. Note that the just intonation constraint is always the top priority of this algorithm, meaning it is not possible to compromise how much any passing chord is in tune using this algorithm.

Research question B was: *can we implement the tuning system from question A so that it models the ideal barbershop quartet?* The algorithm was successfully implemented in C♯, making the answer to question B a definitive yes. Of course, whether this implemented algorithm sounds like the perfect barbershop quartet is a matter of opinion. Nonetheless, since every chord is in just intonation and overall pitch drift can be kept to a minimum, it can be said that this algorithm poses a *mathematically ideal* solution to the tuning problem that is implicitly posed by the Barbershop Harmony Society's Handbook. [5]

The results from Figure 11 show that completely satisfying the four constraints posed in Section 3 is, as predicted, not possible with the proposed algorithm. When *tieRange* and *leadRange* are close to 0, the tie constraint is satisfied, but we land on a different tonal centre than we started on, meaning the drift constraint is not satisfied. With looser ranges, the drift constraint can be satisfied, but there will always be some tied notes that need to be retuned, meaning the tie constraint has to be let go. Different barbershop

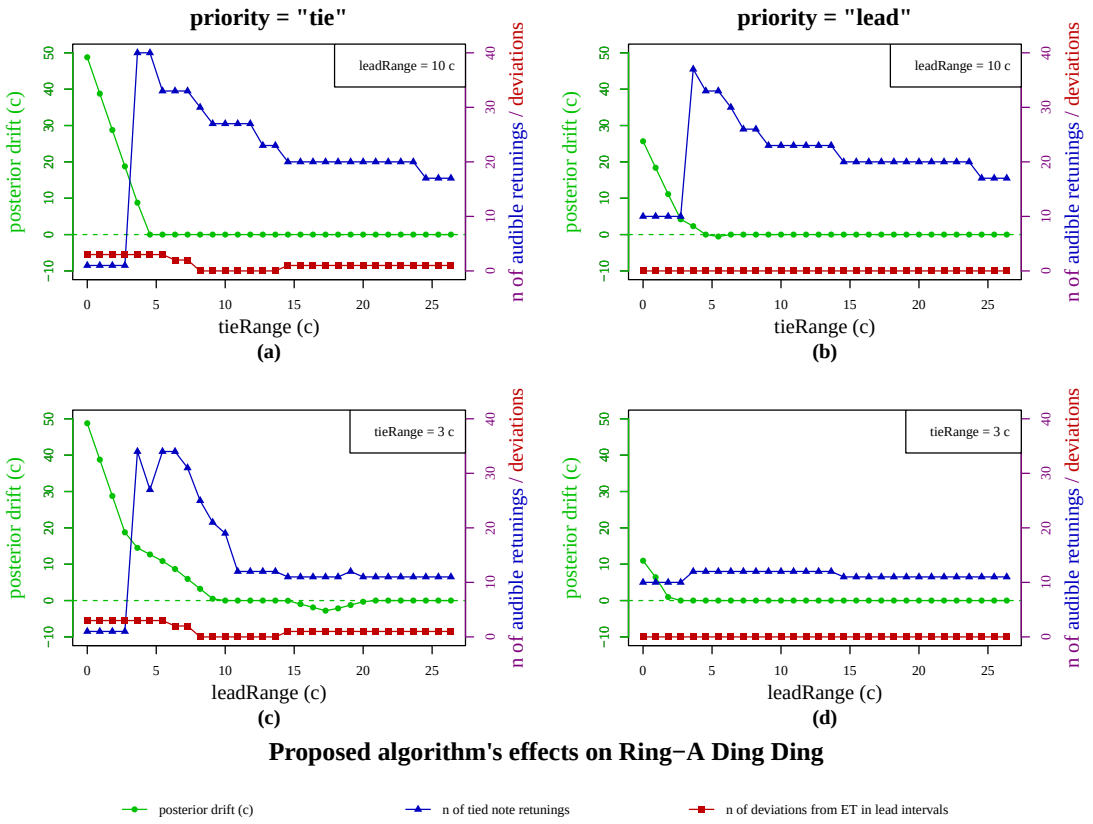



**Figure 11**: Results from the proposed algorithm on the tag of *Ring-a-Ding Ding* (sheet music in Figure 10). The graphs show the effects of parameter changes (x-axis) on three measures (y-axes): overall tonal centre drift in cents (green), number of tied note retunings over 3 cents (blue) and number of deviations from equal temperament in the lead voice larger than 10 cents (red). The tieRange and leadRange parameters determine how loosely the algorithm allows the most important held notes to retune (tie constraint) and how loosely it allows the lead voice's intervals to deviate from equal temperament. The left two graphs have the *priority* set to "tie", whereas the right two graphs have it set to "lead". Both parameters appear to have a similar effect on how many notes need to retune, how many lead intervals are outside equal temperament and how much the song drifts overall.

songs may give different results. *Ring-a-Ding Ding* is a relatively hard song to tune, meaning it might be possible to find a satisfactory tuning for some common barbershop songs. More examples of songs can be found in this thesis' `GitHub repository`.

A few interesting remarks can be made about the results from Figure 11. Firstly, a very low *tieRange* actually causes a very high number of retuned tied notes over 3 cents. That is because the algorithm prioritises tied notes in the order lead-bass-baritone-tenor; if the bass and tenor are both holding a note, then the tenor note might be completely ignored to ensure the bass note does not need to be retuned. Therefore, if the algorithm is very strict about the bass retuning, the tenor will only need to be retuned even more. Secondly, a steady *tieRange* and a *priority* set to lead appears to be a simple way to get a relatively low number of tied note retunings, because the lead interval can always be satisfied. Lastly, when the *priority* is set to "tie", there is a certain area of *tieRange* and *leadRange* values where not a single lead interval has to deviate from equal temperament, even though both lower and higher values for the Range parameters cause some deviations to occur. The exact reason for this dip is unclear, but it most likely also has to do with the effects of the algorithm's priorities.

## 7.2 Musical context

The use of a model such as the one proposed is debatable. For human performers, this algorithm would be impossible to realise; the exact calculations required would be too complicated to carry out while singing and the tiniest intonation changes cannot be replicated using the human voice. Also, the algorithm's goal is not to "capture theories of cognitive functioning in mathematical or computational form", which means it is not exactly a cognitive model. [24] However, the algorithm's results could serve as a reference for real singers to hear what a potentially ideal method of tuning could be. Additionally, high-level findings from the algorithm such as described in section 6 could give singers a frame of reference as to how they should generally tune their notes in a barbershop context.

Even the concept of just intonation itself is a subject of debate: are the intervals from the harmonic series really the best way to tune? Hagerman & Sundberg [25] analysed recordings of barbershop quartets and found that the tenor, baritone and bass use the lead as a reference to adjust their pitches, but their intervals often deviate from just intonation. [26] Other studies have found musical intervals so similarly deviate from just intonation. [27], [28] Thanks to new technologies in music information retrieval, intonation and drift in choirs have been analysed in recent years. [14], [29], [30] Results from analyses like these have even led some researchers to conclude that just intonation as a whole is not the ideal tuning method. [31], [32] However, not enough research has been done to completely isolate these findings from twelve-tone bias. [33]

At the same time, it is important to note that twelve-tone bias does play a role in this article's proposed tuning system. Barbershop music is composed with the western standard division of the octave in mind, which means notes that sound the same on the piano are often meant to sound the same in singing as well. A tuning method that does not adhere to the twelve-tone system and is, therefore, more culturally independent, was proposed by Sethares. [7] Instead of using fractions from the harmonic series, he proposes to analyse a note's timbre using its *dissonance curve*. However, since the present article focuses specifically on barbershop, dissonance curves will always adhere to the natural harmonic series and using the twelve-tone system is commonplace. For these reasons, it was decided not to use Sethares' proposal in the current research.

## 7.3 Philosophical Framework

Philosophy regarding artificial intelligence recognises two classifications of views about the future of the technology: *weak AI* and *strong AI*. Weak AI supposes that AI can merely exist as tools to make human lives easier, whereas strong AI also allows it to learn and understand as humans do and have other

cognitive states. [34] Since music-making is a creative process, crossing the border of outperforming humans in music might be seen as strong AI. [35] However, when a computer is only following a set of rules without being able to add emotion, it might lack the ability to create beauty. [36] The fact that the proposed algorithm in this thesis only follows rules to enrich man-made music explicitly places it in the classification of weak AI.

Tools that allow computers to make music might give rise to a number of ethical concerns which are important to keep in mind during future developments. Firstly, tool-assisted art making may cause an *achievement gap*, in which the connection between human effort and workplace outcome is severed, thereby undermining achievement in the workplace. [37] Secondly, human control over these systems remains important. [38] The UNI Global Union prescribes that AI systems must keep the legal status of tools, meaning the responsibility stays with humans. [39] Lastly, AI systems might take over human jobs, which might have different consequences: either humans get more rewarding tasks, or human jobs completely disappear, creating the need for a universal basic income. [40]

The proposed algorithm alone is not enough to replace human barbershop quartets; its intended use is to serve performers with a suggestion of how to sing their repertoire. However, in combination with tools such as Synthesizer V [41], which can output a human-sounding performance given a score, an automatic tuning system does potentially bring the state-of-the-art one step closer to replacing human singers. Therefore, it is important to keep in mind achievement gaps, human control and the effect on the economy in further research.

## 7.4 Future Work

The input for this algorithm was a newly-defined language, described in Appendix A. Translating a score into this language has turned out to be a tedious process and could be optimised better. One more common way to describe music in plain text is *abc notation*, which also allows chord annotation. [42] However, since automatic chord recognition has improved in the past couple of decades [43], an implementation of this algorithm could easily be built that simply takes MIDI as input and automatically annotates the chords. Essentially, the only thing that the tuning system has to add as output is pitch bend messages.

The scope of this thesis did not allow for a comparison to real-world barbershop quartets. In recent years, systems such as AMPACT can be used to evaluate the tuning tendencies of real choirs. [14], [44] Analysing the difference between mathematically ideal algorithms such as the one proposed and the tendencies of professional barbershop quartets could lead to an even better tuning system. Eventually, a tuning system might be developed that all barbershop musicians can refer to when singing their genre.

## 7.5 Code Availability

All code for the C♯ implementation of the algorithm, along with example songs, source files for the images and TEX-files for this thesis can be found on `https://GitHub.com/teuncb/AdaptiveBarbershop`.

# References

[1] D. D. Boyden, "Prelleur, geminiani, and just intonation," *Journal of the American Musicological Society*, vol. 4, no. 3, pp. 202–219, Oct. 1, 1951, Publisher: University of California Press, ISSN: 0003-0139. DOI: 10.2307/829621. [Online]. Available: https://online.ucpress.edu/jams/article/4/3/202/49422/Prelleur-Geminiani-and-Just-Intonation (visited on 04/10/2023).

[2] J. Fonville, "Ben johnston's extended just intonation: A guide for interpreters," *Perspectives of New Music*, vol. 29, no. 2, pp. 106–137, 1991, Publisher: Perspectives of New Music, ISSN: 0031-6016. DOI: 10.2307/833435. [Online]. Available: https://www.jstor.org/stable/833435 (visited on 04/10/2023).

[3] V. Persichetti, *Twentieth-Century Harmony: Creative Aspects And Practice*, 4th ed. Ww Norton & Co, 1961.

[4] J. Van de Craats, *De fis van Euler. Een nieuwe visie op de muziek van Schubert, Beethoven, Mozart en Bach.* Aramith Uitgevers, 1989.

[5] Barbershop Harmony Society, "Contest and judging handbook," July 2022 2022. [Online]. Available: https://www.barbershop.org/contests/contests-judging.

[6] G. Averill, "Bell tones and ringing chords: Sense and sensation in barbershop harmony," *The World of Music*, vol. 41, no. 1, pp. 37–51, 1999, Publisher: [Florian Noetzel GmbH Verlag, VWB - Verlag für Wissenschaft und Bildung, Schott Music GmbH & Co. KG, Bärenreiter], ISSN: 0043-8774. [Online]. Available: https://www.jstor.org/stable/41700111 (visited on 04/10/2023).

[7] W. A. Sethares, "Adaptive tunings for musical scales," *The Journal of the Acoustical Society of America*, vol. 96, no. 1, p. 10, Apr. 5, 1994, Publisher: Acoustical Society of AmericaASA, ISSN: 0001-4966. DOI: 10.1121/1.410471. [Online]. Available: https://asa.scitation.org/doi/abs/10.1121/1.410471 (visited on 02/13/2023).

[8] "Adaptive tunings," in *Tuning, Timbre, Spectrum, Scale*, W. A. Sethares, Ed., London: Springer, 2005, pp. 155–178, ISBN: 978-1-84628-113-6. DOI: 10.1007/1-84628-113-X_8. [Online]. Available: https://doi.org/10.1007/1-84628-113-X_8 (visited on 04/18/2023).

[9] W. Mohrlok, *The hermode tuning system*, 2003. [Online]. Available: https://sethares.engr.wisc.edu/paperspdf/hermode.pdf.

[10] D. Volkov. "Pivotuner." (2022), [Online]. Available: https://midi.org/component/zoo/item/pivotuner (visited on 02/13/2023).

[11] Utrecht University. "Human-centered Artificial Intelligence - Universiteit Utrecht." (Feb. 10, 2023), [Online]. Available: https://www.uu.nl/onderzoek/human-centered-artificial-intelligence (visited on 02/13/2023).

[12] A. Forte, *Tonal Harmony in Concept and Practice*, 3rd ed. Holt, Rinehart, and Winston, 1979, ISBN: 0-03-020756-8.

[13] J. M. Dougherty, *Choral tuning: A solution to "the question of the supertonic" in just intonation*, Master's thesis, 2004.

[14] J. Devaney, M. Mandel, and I. Fujinaga, "A study of intonation in three-part singing using the automatic music performance analysis and comparison toolkit (AMPACT)," *Proceedings of the 13th International Society for Music Information Retrieval Conference, ISMIR 2012*, pp. 511–516, Jan. 1, 2012.

[15] D. L. Code, "Groven.max: An adaptive tuning system for MIDI pianos," *Computer Music Journal*, vol. 26, no. 2, pp. 50–61, 2002, Publisher: The MIT Press, ISSN: 0148-9267. [Online]. Available: https://www.jstor.org/stable/3681456 (visited on 02/13/2023).

[16] S. Schwär, S. Rosenzweig, and M. Müller, "A DIFFERENTIABLE COST MEASURE FOR INTONATION PROCESSING IN POLYPHONIC MUSIC," *Proceedings of the 22nd ISMIR Conference*, 2021.

[17] A. Horner and L. Ayers, "Common tone adaptive tuning using genetic algorithms," *The Journal of the Acoustical Society of America*, vol. 100, no. 1, pp. 630–640, Jul. 1, 1996, ISSN: 0001-4966. DOI: 10.1121/1.415887. [Online]. Available: https://doi.org/10.1121/1.415887 (visited on 04/21/2023).

[18] L. Garnett, "Ethics and aesthetics: The social theory of barbershop harmony," *Popular Music*, vol. 18, no. 1, pp. 41–61, Jan. 1999, Publisher: Cambridge University Press, ISSN: 1474-0095, 0261-1430. DOI: 10.1017/S0261143000008722. [Online]. Available: https://www.cambridge.org/core/journals/popular-music/article/abs/ethics-and-aesthetics-the-social-theory-of-barbershop-harmony/88DD765DDACCB632A1B5086885B393DE (visited on 02/13/2023).

[19] MIDI Manufacturers Association, *The complete MIDI 1.0 detailed specification*, 2014. [Online]. Available: https://www.midi.org/specifications/midi1-specifications/general-midi-specifications/general-midi-1.

[20] W. B. de Haas. "Music information retrieval based on tonal harmony." Accepted: 2012-02-17T08:19:17Z ISBN: 9789039357354 Publisher: Utrecht University. (Feb. 28, 2012), [Online]. Available: https://dspace.library.uu.nl/handle/1874/226713 (visited on 04/21/2023).

[21] W.-Y. Hsiao, J.-Y. Liu, Y.-C. Yeh, and Y.-H. Yang, *Compound word transformer: Learning to compose full-song music over dynamic directed hypergraphs*, Jan. 7, 2021. DOI: 10.48550/arXiv.2101.02402. arXiv: 2101.02402[cs,eess]. [Online]. Available: http://arxiv.org/abs/2101.02402 (visited on 04/21/2023).

[22] M. Dobroselsky, *Melanchall/drywetmidi*, original-date: 2017-03-22T14:35:05Z, Apr. 14, 2023. [Online]. Available: https://github.com/melanchall/drywetmidi (visited on 04/19/2023).

[23] DominikSchaller. "Barbershop tags - ring-a-ding ding!" (2015), [Online]. Available: https://www.barbershoptags.com/tag-3064-Ring-a-Ding-Ding! (visited on 04/07/2023).

[24] L. van Maanen and S. Miletić, "The interpretation of behavior-model correlations in unidentified cognitive models," *Psychonomic Bulletin & Review*, vol. 28, no. 2, pp. 374–383, Apr. 1, 2021, ISSN: 1531-5320. DOI: 10.3758/s13423-020-01783-y. [Online]. Available: https://doi.org/10.3758/s13423-020-01783-y (visited on 04/03/2023).

[25] B. Hagerman and J. Sundberg, *Fundamental frequency adjustment in barbershop singing*. Citeseer, 1980.

[26] S. E. Abbott, *Acoustic evaluation and analysis of the female barbershop tenor voice*. The Florida State University, 2001.

[27] J. Nordmark and S. Ternström, "Intonation preferences for major thirds with non-beating ensemble sounds," Mar. 1, 1996.

[28] J. Sundberg, A. Friberg, and L. Frydén, "Rules for automated performance of ensemble music," *Contemporary Music Review*, Aug. 24, 2009, Publisher: Taylor & Francis Group, ISSN: 1029-0087. DOI: 10.1080/07494468900640071. [Online]. Available: https://www.tandfonline.com/doi/abs/10.1080/07494468900640071 (visited on 04/21/2023).

[29] M. Mauch, K. Frieler, and S. Dixon, "Intonation in unaccompanied singing: Accuracy, drift, and a model of reference pitch memory," *The Journal of the Acoustical Society of America*, vol. 136, p. 401, Jul. 1, 2014. DOI: 10.1121/1.4881915.

[30] J. Dai and S. Dixon, "Intonation trajectories within tones in unaccompanied soprano, alto, tenor, bass quartet singing," *The Journal of the Acoustical Society of America*, vol. 146, pp. 1005–1014, Aug. 1, 2019. DOI: 10.1121/1.5120483.

[31] R. Parncutt and G. Hair, "A psychocultural theory of musical interval: Bye bye pythagoras," *Music Perception*, vol. 35, no. 4, pp. 475–501, Apr. 1, 2018, ISSN: 0730-7829. DOI: 10.1525/mp.2018.35.4.475. [Online]. Available: https://doi.org/10.1525/mp.2018.35.4.475 (visited on 04/03/2023).

[32] R. Kopiez, "Intonation of harmonic intervals: Adaptability of expert musicians to equal temperament and just intonation," *Music Perception*, vol. 20, no. 4, pp. 383–410, Jun. 1, 2003, Publisher: University of California Press, ISSN: 0730-7829. DOI: 10.1525/mp.2003.20.4.383. [Online]. Available: https://online.ucpress.edu/mp/article/20/4/383/62149/Intonation-of-Harmonic-Intervals-Adaptability-of (visited on 04/10/2023).

[33] E. M. Burns, "7 - intervals, scales, and tuning," in *The Psychology of Music (Second Edition)*, ser. Cognition and Perception, D. Deutsch, Ed., San Diego: Academic Press, Jan. 1, 1999, pp. 215–264, ISBN: 978-0-12-213564-4. DOI: 10.1016/B978-012213564-4/50008-1. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B9780122135644500081 (visited on 04/21/2023).

[34] J. R. Searle, "Minds, brains, and programs," *Behavioral and Brain Sciences*, vol. 3, no. 3, pp. 417–424, Sep. 1980, Publisher: Cambridge University Press, ISSN: 1469-1825, 0140-525X. DOI: 10.1017/S0140525X00005756. [Online]. Available: https://www.cambridge.org/core/journals/behavioral-and-brain-sciences/article/abs/minds-brains-and-programs/DC644B47A4299C637C89772FACC2706A (visited on 04/20/2023).

[35] J. Gottschall, "The rise of storytelling machines," *What to Think about Machines that Think; Brockman, J., Ed*, pp. 179–180, 2015.

[36] A. Braga and R. K. Logan, "The emperor of strong AI has no clothes: Limits to artificial intelligence," *Information*, vol. 8, no. 4, p. 156, Dec. 2017, Number: 4 Publisher: Multidisciplinary Digital Publishing Institute, ISSN: 2078-2489. DOI: 10.3390/info8040156. [Online]. Available: https://www.mdpi.com/2078-2489/8/4/156 (visited on 04/20/2023).

[37] J. Danaher and S. Nyholm, "Automation, work and the achievement gap," *AI and Ethics*, vol. 1, no. 3, pp. 227–237, Aug. 1, 2021, ISSN: 2730-5961. DOI: 10.1007/s43681-020-00028-x. [Online]. Available: https://doi.org/10.1007/s43681-020-00028-x (visited on 04/20/2023).

[38] J. Fjeld, N. Achten, H. Hilligoss, A. Nagy, and M. Srikumar, *Principled artificial intelligence: Mapping consensus in ethical and rights-based approaches to principles for AI*, Rochester, NY, Jan. 15, 2020. DOI: 10.2139/ssrn.3518482. [Online]. Available: https://papers.ssrn.com/abstract=3518482 (visited on 04/20/2023).

[39] UNI Global Union, *Top 10 principles for ethical AI*, 2017.

[40] R. Calo, "Artificial intelligence policy: A primer and roadmap," *U.C. Davis Law Review*, vol. 51, p. 399, 2017. [Online]. Available: https://heinonline.org/HOL/Page?handle=hein.journals/davlr51&id=413&div=&collection=.

[41] K. Hua. "Synthesizer V — Dreamtonics," Dreamtonics. (Apr. 15, 2020), [Online]. Available: https://dreamtonics.com/synthesizerv/ (visited on 04/20/2023).

[42] C. Walshaw, *ABC2mtex: An easy way of transcribing folk and traditional music, version 1.0*, University of Greenwich, London, 1997.

[43] J. Burgoyne, L. Pugin, C. Kereliuk, and I. Fujinaga, *A Cross-Validated Study of Modelling Strategies for Automatic Chord Recognition in Audio*. Jan. 1, 2007, 251 pp., Pages: 254.

[44] P. Chandna, H. Cuesta, D. Petermann, and E. Gómez, "A deep-learning based framework for source separation, analysis, and synthesis of choral ensembles," *Frontiers in Signal Processing*, vol. 2, p. 808 594, Apr. 1, 2022. DOI: 10.3389/frsip.2022.808594.

[45] "Pitch notation conventions," in *A History of Stringed Keyboard Instruments*, S. Pollens, Ed., Cambridge: Cambridge University Press, 2022, pp. xxi–xxii, ISBN: 978-1-108-42199-7. [Online]. Available: https://www.cambridge.org/core/books/history-of-stringed-keyboard-instruments/pitch-notation-conventions/BA088BB4AB776A5CF7B6975AAB3DB843 (visited on 04/21/2023).

[46] C. Dobrian. "Timing in MIDI files — computer audio and music programming – 2014." (May 19, 2014), [Online]. Available: https://sites.uci.edu/camp2014/2014/05/19/timing-in-midi-files/ (visited on 04/21/2023).

The cover art and figures including piano keyboards were designed using assets from Freepik.com.

# Appendices

These appendices describe the language in which `.txt` files should be written for the presented C♯ program to interpret music. Appendix A describes how a Barbershop score can be converted into a `.txt` file. Appendix B then describes how a user can prescribe just intonation frequency ratios based on a playing chord.

## A Input Language for Songs

The C♯ implementation of the proposed algorithm takes `.txt` files containing tagged scores as input. Converting a score to a `.txt` file involves splitting it into bits of time where no notes change called `Chord`s and interpreting the playing harmonies to add chord symbols. Figure 6 shows an example of the translation process. Some more examples of converted scores can be found in the `Songs folder on the GitHub repository`.

A note consists of exactly four characters: a note name, an accidental, an octave, and a tie indication. If a voice is silent in a `Chord`, the note should simply be four spaces. The note name is simply a lower-case letter between a and g. The accidental is a character from the set $\{\texttt{\#}, \texttt{b}, \texttt{n}\}$, which stand for "sharp", "flat" and "natural", respectively. The octave is a number between 0 and 8, written as in Scientific Pitch Notation. [45] The octave makes no difference in the tuning process, but the note must be present on an 88-key piano. The tie indication is either "`t`" or a space, which indicates whether this note keeps playing in the next `Chord`. For example, "`an3 `" would be the highest A below middle C that stops after this `Chord`, whereas "`e#4t`" would be the first E♯ (or, in this case equivalently, F) above middle C, that keeps playing after this `Chord`. In the last example, the next `Chord` should also contain this E♯ or F in the same voice.

A chord symbol consists of exactly three characters: a note name, an accidental and a chord type. The note name and accidental work the same as for notes, though in a chord symbol the note name should be in upper-case. The chord type is a character from the set $\{\texttt{M}, \texttt{m}, \texttt{7}, \texttt{o}, \texttt{0}\}$, which stand for major, minor, dominant seventh, diminished and half diminished, respectively. The chord type decides which `Fractions` file is chosen to tune the notes in the current `Chord`. The default algorithm uses the same file for major and dominant chords and also uses the same file for minor and half-diminished chords.

A `Chord` is described using a single line containing a chord symbol, four notes (enclosed by brackets and separated by commas), a duration and optional comments after a `%`. The notes are ordered bass - baritone - lead - tenor. Like the octave, the duration is only used for MIDI output and has no effect on tuning. The algorithm always uses the standard [46] timecode of 480 ticks per quarter note, meaning an eighth note would be $0.5 \cdot 480 = 240$ ticks.

# B Input Language for Fractions

A `.txt` file describing interval ratios consist of exactly twelve lines, one for each of the twelve possible notes in the input. Each line consists of a frequency ratio with the root and additional comments after a `%`. The first line corresponds to the chord root and is therefore logically `1/1`. The next lines each count up from the root in semitones. The eighth line therefore corresponds to a fifth, meaning it would most likely be `3/2` (see 2.1). The `TuningTables folder on the GitHub repository` contains some examples of `Fraction` files.