AutoScale: Automatic and Dynamic Scale Selection for Live Jazz Improvisation

Thibault Jaccard
thibault.jaccard@gmail.com

Robert Lieck
robert.lieck@epfl.ch

Martin Rohrmeier
martin.rohrmeier@epfl.ch

Digital and Cognitive Musicology Lab
École Polytechnique Fédérale de Lausanne
1015 Lausanne, Switzerland

ABSTRACT

Becoming a practical musician traditionally requires an extensive amount of preparatory work to master the technical and theoretical challenges of the particular instrument and musical style before being able to devote oneself to musical expression. In particular, in jazz improvisation, one of the major barriers is the mastery and appropriate selection of scales from a wide range, according to harmonic context and style.

In this paper, we present AutoScale, an interactive software for making jazz improvisation more accessible by lifting the burden of scale selection from the musician while still allowing full controllability if desired. This is realized by implementing a MIDI effect that dynamically maps the desired scales onto a standardized layout. Scale selection can be pre-programmed, automated based on algorithmic lead sheet analysis, or interactively adapted.

We discuss the music-theoretical foundations underlying our approach, the design choices taken for building an intuitive user interface, and provide implementations as VST plugin and web applications for use with a Launchpad or traditional MIDI keyboard.

Author Keywords

music theory, user interface, MIDI scale mapper

CCS Concepts

•Applied computing → Sound and music computing; Performing arts; **•Human-centered computing → User interface programming;**

1. INTRODUCTION

Musical improvisation requires both a thorough understanding of the underlying harmonic structure as well as a high level of technical expertise to employ this understanding in actual improvisation. This generally requires many years of studies with skilled teachers and hundreds and thousands of hours of laboriously practicing every scale in every key. This is in particular the case for jazz, where many styles have a rich repertoire of scales and chord progressions that frequently contain modulations and substitutions, but it

Figure 1: The pedal harp[1]

equally applies to other musical genres, like pop or classical music. This extensive amount of preparatory work precludes many non-professionals from following their musical inspirations and puts a high barrier before fulfilling and gratifying musical expression.

The goal of our work is to lower this burden of theoretical and technical prerequirements and make jazz improvisation more accessible to a wider audience. To this end we employ two principles, partial *automatization* and an intuitive *user interface design*. In particular, the combination of both approaches offers new possibilities that go beyond what is possible with traditional "static" instruments.

The technical difficulty of utilizing different scales varies among traditional instruments. On the piano, musicians have to practice scales as distinct patterns of white and black keys. Before being able to express herself freely, a pianist has to master any possible scale, such that her fingers can run intuitively on the keyboard. Alternatively, the guitar offers some kind of isomorphism, in the sense that what the musician learns in one key can be reused in other keys. Unlike the piano, transposing a melody on the guitar simply means playing the same pattern somewhere else on the neck. An even better example of an isomorphic instrument is the pedal harp. On the harp, there is one string per diatonic degree and per octave and alterations of the scale are set by seven pedals, one for each degree. Once the pedals are adjusted, the scale is fixed and the musician can freely play the strings while always remaining in the correct scale.

The guiding idea behind this work was the following: *What if we had an instrument like the harp, where it is impossible to play "out of scale", but where additionally scale alterations could be automated to follow a specific chord progression?* Instead of working with an acoustic harp, we decided to use modern technology, and create a new kind of MIDI scale mapper. Instead of being static like Ableton's

[1] `https://commons.wikimedia.org/wiki/File:Salvi_harp_Diana`

NIME'20, July 21-25, 2020, Royal Birmingham Conservatoire, Birmingham City University, Birmingham, United Kingdom.

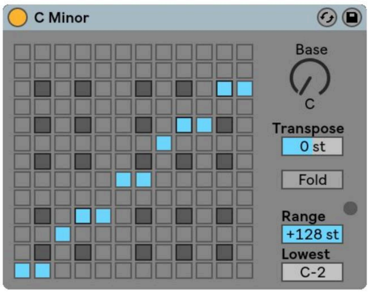

Figure 2: Ableton scale plugin mapping the C major scale onto C natural minor

native MIDI scale mapper (Figure 2), our version promotes alterations and scale changes and can be automated following a chord progression.

We wanted the user interface to be intuitive enough for anybody to be able to interact with it. Even without any music theory background, anybody should be able to modify the proposed scales and explore the harmonic possibilities. However, the system is adaptable enough to let experienced musicians use their skills without constraints.

The scale selection is either automated based on a chord progression or done manually using an interactive chromatic circle. The former allows the user to play on any jazz tune without struggling with the question which scale to play in, while the latter lets them explore and compose freely.

The playing is done using an isomorphic layout, such as the piano white keys or a pad matrix, where each key (or pad) is associated with a degree within the scale. The player cannot, therefore, play out of the scale, but they can change the scale at will. The system acts as a MIDI effect, or more precisely as a dynamic MIDI scale mapper.

Our system extends the range of tools for interactive human-machine improvisation [4, 5, 10] and adds to the methods for representing musical scales [11, 7, 1, 18].

In the remainder of this paper, we will first discuss the theoretical model used for this project in Section 2 and then present our user interface in Section 3, including the employed music-theoretical representation (Section 3.1) and details on our implementation in two prototype applications (Section 3.2).

2. MODEL

2.1 Scale Theory

The system presented here is based on three assumptions:

- **Octave equivalence**

  Any scale can be characterized by the notes present in one octave and every octave contains the same notes. A note in one octave is equivalent to the corresponding notes in all other octaves.

- **12 tone equal temperament**

  The octave is divided into 12 equally distant intervals, called semitones. The model may be extended later to differentiate enharmonic notes, but for clarity sake, we will stick to the pitch class model [2, p.776] in this paper.

- **7 degrees nomenclature**

  In addition to the pitch class, every note is characterized by a degree, which corresponds to the conventional names in music theory (C = 1, D = 2, ...) [12, 17]. By default, notes are natural (♮, C major scale), but they can be altered at will (♭ = −1 semitone, ♯ = +1 st, ♭♭ = -2 st, ♯♯♯ = +3 st, ...) to match any pitch class.

name	degree	pitch class
C	1	0
C♯		1
D♭	2	
D		2
D♯		3
E♭	3	
E		4
F	4	5
F♯		6
G♭	5	
G		7
G♯		8
A♭	6	
A		9
A♯		10
B♭	7	
B		11

Figure 3: Nonexhaustive list of notes and their notation in degree and semitone

Given those rules, any note is represented by two integers, namely the pitch class $p \in \{0, \ldots, 11\}$ and the degree $d \in \{1, \ldots, 7\}$. Figure 3 presents a non-exhaustive list of notes.

To find the alteration (♭, ♯) of a note N_0 given its degree and pitch class (d_0, p_0) we can therefore compare it to the C major scale. As the latter contains a note for each degree, a note with the same degree but possibly different pitch $N_1 = (d_1 = d_0, p_1)$ exists within the C major scale, and the alteration of N_0 can be calculated as

$$a_0 = (p_0 - p_1 + 6) \bmod 12 - 6\ . \tag{1}$$

This method only works to calculate alterations between −6 and +5 semitone steps.

Our model is not limited to the diatonic case. Generally, scales may contain more than one note with the same degree or even miss a degree. For instance, the C altered scale is considered to have twice the second degree and no fifth degree [6, p. 71]: (1, 0), (2, 1), (2, 3), (3, 4), (4, 6), (6, 8), (7, 10) with notes being represented by (degree, pitch class), as described above.

One advantage of using this model is the full determination of intervals between notes. Not only the number of semitones can be calculated, but also the distance in terms of degrees. This is advantageous, for instance, for employing unequal temperament tunings [3] using the MPE protocol to fine tune each note according to its degree. The interval between $N_0 = (d_0, p_0)$ and $N_1 = (d_1, p_1)$ is calculated as follows:

$$I = ((d_1 - d_0) \bmod 7 + 1, (p_1 - p_0) \bmod 12) = (d_i, p_i) \tag{2}$$

The names of the intervals are the classical ones [6, p.3]. Figure 4 presents a non-exhaustive list of intervals.

Considering major and perfect as being the natural alterations (= 0, so minor = −1, augmented = +1, ...), intervallic alterations can be calculated the same way as notes alterations, by referring to the major scale beginning on the initial note. Indeed, the major scale only contains major or perfect intervals. Essentially, interval names are defined in the same way as notes names, with the only difference that intervals are relative while notes are absolute.

For example, the interval between the note C (1, 0) and E♭ (3, 3) is $((3-1) \bmod 7+1, (3-0) \bmod 12) = (3, 3)$ which is a minor third, while between C (1, 0) and D♯ (2, 3) is

name	degree	semitone
perfect unison	1	0
augmented unison		1
minor second	2	
major second		2
augmented second		3
minor third	3	
major third		4
perfect fourth	4	5
augmented fourth		6
diminished fifth	5	
perfect fifth		7
augmented fifth		8
diminished sixth	6	
major sixth		9
augmented sixth		10
minor seventh	7	
major seventh		11

Figure 4: Non-exhaustive list of intervals and their notation in degree and semitone intervals

type	core notes			function(s)
	3rd	5th	7th	
augmented	maj	aug	maj	augmented
major	maj	per	maj	tonic, subdominant or other
dominant	maj	-	min	dominant
minor	min	per	-	tonic, subdominant or other
half-diminished	min	dim	min	subdominant or other
diminished	min	dim	dim	dominant (dim)

Figure 5: Chord types and harmonic functions

$((2-1) \bmod 7 + 1, (3-0) \bmod 12) = (2, 3)$, we get an augmented second. It is interesting to note that when looking at intervals from C, the intervallic alteration is the same as the note's alteration. In this case, $C \rightarrow E\flat$ ($\flat = -1$) is a minor third (minor = -1), while $C \rightarrow D\sharp$ ($\sharp = +1$) is an augmented second (augmented = +1).

The intervallic relations of interest here are those characterizing the distance between the root note and any other note within the scale. By looking at intervallic alterations of every note of a heptatonic scale, we can easily compare it to the major scale. As we said earlier, the latter only has natural intervallic alterations (major or perfect). Any intervallic alteration added will highlight the difference between the obtained scale and the major one. For example, the intervallic alterations of mixolydian are 0, 0, 0, 0, 0, 0, -1, and we can directly see that this scale can be obtained by altering down the seventh of the major scale starting on the same root note.

Given this model, chords and scale are both considered as subsets of notes. For instance the D minor triad Dm = (2, 2), (4, 5), (6, 9) has the intervallic alterations 0, -1, and 0, respectively.

2.2 Chord Types

In order to relate scales to chords when parsing a chord progression, we organize chords into six categories, which we will call types. Those are presented in Figure 5. They correspond to those used in Schoenberg's *Theory of Harmony* [16].

The middle three columns show which notes are used to identify the chord/scale type. As we focus on jazz chord progressions [13, 14], the dominant chords always contain the minor 7th. A major chord with no 7th is therefore never considered a dominant chord.

The last column presents the possible corresponding chord functions for each type. As it can be seen, some chord types correspond to a single chord function (dominant = dominant) while other gather multiple chord functions (major, minor). In this sense, the chord type is a more general chord characterization than the chord function.

Based on a given chord type, we can derive a list of matching scales that contain the same core notes. Most of the scales within a type's list are highly interchangeable and the user should be able to switch between them at will. Figure 6 lists the scales for each chord type.

By default, the chosen scale for each chord type is the one with an asterisk. In general, the appropriate scale does not only depend on the chord type but to some extend also on the harmonic context. This is for instance the case for the minor chord type, which therefore has two different default scales. The dorian scale is appropriate for the case where the minor chord is subdominant, while the melodic minor scale is used on a minor tonic chord. To differentiate these two cases, the harmonic context is considered: If the minor chord is followed by a dominant chord, we analyse whether it is part of a ii-V progression (or one of the possible substitution ([6, p.81], [14]): iv-V, ♭vi-V, vii-V) or not. If it is, the minor chord is interpreted as a subdominant chord and dorian is chosen as the default scale. Otherwise, if the minor chord is not part of a ii-V progression (or one of the possible substitutions), it is interpreted as a minor tonic chord and, therefore, the melodic minor scale is used as the default scale. This approach can be considered as first heuristic and the model can be improved in the future.

type	scales
augmented	augmented lydian
	augmented ionian*
major	harmonic major
	ionian*
	lydian
	lydian ♯9
dominant	whole tone
	lydian dominant
	diminished 1/2 1
	altered dominant
	mixo. ♭9 ♭13
	mixolydian ♭13
	mixolydian*
minor	dorian ♯11
	dorian*
	dorian ♭9
	phrygian
	aeolian
	harmonic minor
	melodic minor*
half-diminished	altered
	half diminished*
	locrian
	locrian ♯13
diminished	diminished 1 ½*
	diatonic diminished

Figure 6: Scale list for each chord type

i	ii	iii	iv	v	vi	vii

Figure 7: Color code for relative degrees

degree (interval)	relative alteration symbol			
	--	-		+
2nd, 3rd, 6th and 7th	diminished	minor	major	augmented
4th and 5th		diminished	perfect	augmented

Figure 8: Alterations representation

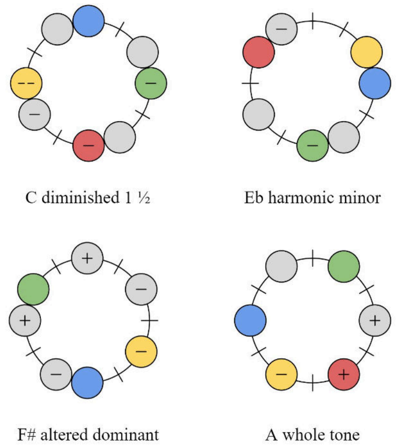

Figure 9: Four scales represented in chromatic circles

3. USER INTERFACE

3.1 Representation

3.1.1 Colors

Colors are well suited to draw attention to specific locations of the user interface and convey categorical information. A color code therefore seemed ideal for representing specific tonal functions, irrespective of the concrete geometric layout they appear in. The colors used here (Figure 7) are a suggestion that may be changed according to the user's preferences. Central to the color coding scheme is that colors are always determined relative to the current root note, that is, they represent intervals relative to the root and not absolute tones. Colors are also used to identify scale degrees. The proposed coloring scheme highlights the root, third, fifth and seventh of any scale, independently of the alterations (major, minor, perfect, diminished etc.). If other notes are added (e.g. for octatonic scales), they will be gray.

3.1.2 Alterations

Intervallic alterations are represented by plus and minus signs ($-$, $+$). We chose major and perfect as the "neutral" alterations and only indicate deviations to keep the representation clear. The reason for using mathematical signs as opposed to music theoretic notation for alterations (♭, ♮, ♯) was that these are more widely known in a general audience. The symbols are summarized in Figure 8.

3.1.3 Chromatic Circle

Given our assumptions of octave equivalence and the 12 tone equal temperament, a well-established way to represent scales is the chromatic circle [9]. Hence for the user interface, we chose a chromatic circle with the note C on top and semitones incrementing clockwise. Figure 9 reveals four scale examples represented using our system.

This representation aims to be as general as possible. Indeed, any scale that can be played on a piano can be represented on the chromatic circle. Moreover, our version of the chromatic circle contains information about degrees and intervallic alterations. It can, therefore, be considered as an enhancement of the standard chromatic circle representation.

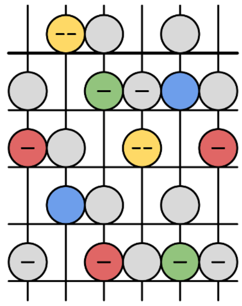

Figure 10: C diminished 1 $^1/_2$ on a guitar

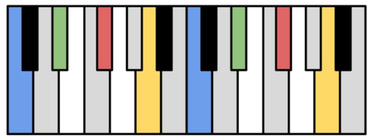

Figure 11: C diminished 1 $^1/_2$ on a piano

3.1.4 Other Representations

One of the advantages of using colors is that correspondence can easily be made with other geometrical representations. Figure 10 shows a scale represented on a guitar and Figure 11 on a piano layout.

These representations are not yet present in our software at this stage, but they can easily be implemented to show the correspondence with any type of instrument. This can also help with learning how to read the chromatic circle.

3.2 Implementation

We implemented our approach in two prototypes, a VST plugin, and a web-based prototype. The former has been coded in C++ using the JUCE framework [15], the second one using the p5.js library [8]. Both consist of the same two components: a chord progression visualizer and an interactive chromatic circle. In the web-based prototype this is implemented in two separate apps communicating through MIDI, one for the progression and one for the chromatic circle. The plugin version is able to parse chord progressions in the iReal Pro MusicXML format, while the web app uses simple text files containing chords in their classic notation (Am7, C7b9, etc).

3.2.1 Mouse/Touchsreen

The interactive chromatic circle lets the user obtain and modify any scale. Starting from a given scale, the user can alter a degree by simply dragging the corresponding circle onto a free spot near it. When developing and testing the interface, this provided the most intuitive way of modifying scales. When the note is released, its alteration sign is automatically updated. When moving notes in this way, the colors remain the same because notes retain their respective scale degree and only change their alteration. For example, a third can be altered, but it will remain a third. The user can, for example, move the third from a major scale down a semitone (drag it $\pi/6$ rad anticlockwise) to obtain a melodic minor scale.

To only change the fundamental of a scale, that is, to switch to a different mode that uses the same notes but has

rel. pitch class	rel. degree
0	1
1	2
2	
3	3
4	
5	4
6	
7	5
8	6
9	
10	7
11	

Figure 12: Relative degrees of new notes according to their relative pitch classes

a different fundamental (e.g. from major to dorian, phrygian etc.), the user can click or tap on a non-fundamental note. This is a great way to discover which are the intervallic alterations of all modes from a scale.

New notes can be added by clicking or tapping on a free spot and deleted by dragging them out of the circle. When adding a note, its degree is defined by the following rules:

- If the neighboring notes have a distance of two in terms of their degree, the new note takes the degree in between.

- For all cases, the degree is defined by the relation shown in Figure 12 with the pitch class being computed relative to the root note of the current scale. The degree in this table (d_r) is also relative to the root's degree (d_f), which means that the absolute degree (C, D, ...) is found using the following formula:

$$d = ((d_f + d_r - 2) \bmod 7) + 1 \ . \tag{3}$$

The option to add and delete notes is currently only available in the plugin version, while the web-based prototype is limited to heptatonic scales.

3.2.2 *Piano Keyboard*

When using heptatonic scales, a MIDI piano keyboard can be conveniently used to play within those scales. The system then acts as a standard MIDI scale mapper and the white keys are associated with corresponding degrees of the scale. Our app, therefore, has a MIDI input and a MIDI output, where the input only accepts natural notes (white keys), while the output corresponds to the selected scale.

If a key is held during an alteration process (note on sent but not note off), the pitch class is remembered by the software in order to send the correct note off message when the key is released.

This system works well for most cases, but is still limited as octatonic scales cannot be intuitively mapped on the piano keyboard. An alternative solution is discussed in the next section.

3.2.3 *RGB Pad Matrix*

Pad matrices, such as the Launchpad Pro, the ROLI Lightpad or the Native Instrument Maschine, offer a better layout as they are not restricted to seven degrees and mostly have RGB capabilities. Therefore, a stronger relation can be made between the instrument and the scale representation.

Since the interval between rows can be defined arbitrarily we tested different layouts. For guitarists, an interval of a fourth between the rows corresponds best to what they are used to, while for violinists a fifth is more intuitive. For this

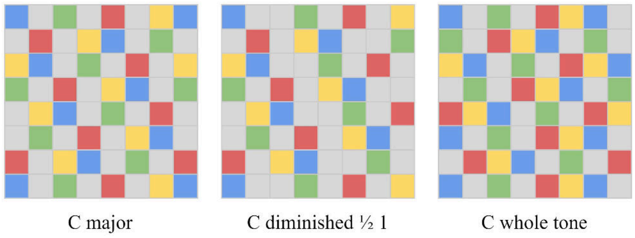

Figure 13: Launchpad layout of three scales, with a fifth between rows

reason, we did not fix a specific interval but this parameter can be user-defined. An example of the layout of three different scales is shown in Figure 13. Note that any diatonic scale with C as its root is represented by the same layout as C major (Figure 13, first layout). This is because only the degrees are shown but not the corresponding alterations.

One of the great advantages of using an isomorphic instrument is that all the techniques and reflexes learned in one scale can be applied in any other scale and in any key. Furthermore, when using only heptatonic scales (as in the web app prototype), distances between the pads are always related to the interval in terms of degrees only. This allows musicians to focus more on the development of melodic skills and allows their application in a wide range of harmonic situations.

Our prototypes have been tested using the Launchpad Pro. As the web app is restricted to heptatonic scales, it can be used both with a Launchpad Pro and with a standard MIDI piano controller.

3.2.4 *Progression*

Chords are parsed by converting them into arrays of notes. Their type is then determined using the rules presented in Figure 5. Based on the type, a group of scales is associated to each chord, as shown in Figure 6. The user may preselect a scale from the list by using the up/down arrow keys. Chords may either be selected manually by using the left/right arrow keys or be automatically advanced by using a MIDI clock signal. This allows to use chord progression when recording over DAW tracks or in conjunction with hardware MIDI clock sources like sequencers.

In the web based prototype, which consists of two separate applications (progression and chromatic circle, see above), the scale information is sent over MIDI using a virtual MIDI cable. In this way, only the progression app needs to analyze the chords, determines the type, and hold the scale list. When a new chord is selected or the user changes the scale, the scale information is sent to the chromatic circle app in order to update the display according to the new scale.

To gain a better intuition we discuss a brief example. We will use a simple progression of four chords: Cmaj7, A7, Dm7, G7. For the first chord, the user keeps the default scale, C ionian. On the second chord, they change the default A mixolydian to A mixolydian ♭9 ♭13. The third chord is played using the D dorian ♭9 scale, and on the last chord the much more dissonant G altered dominant scale is chosen. Figure 14 shows the state of the progression (top), the chromatic circle (middle) and the Launchpad (bottom) for each chord in the progression (left to right).

4. CONCLUSION

We have presented a new kind of responsive scale representation that facilitates improvisation over complex chord

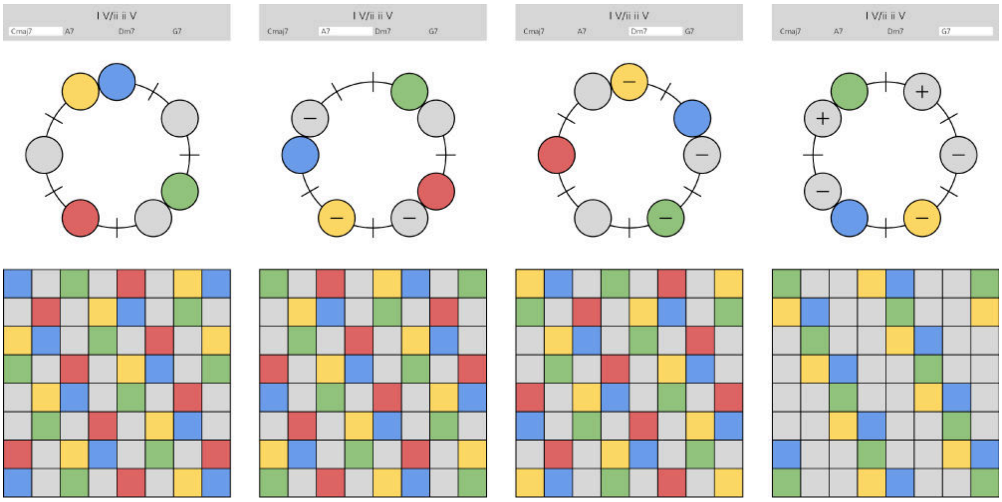

Figure 14: Demonstration of a short chord progression (Cmaj7, A7, Dm7, G7) showing the chord itself (top), the chromatic circle (middle), and the launchpad (bottom) for each chord

progressions. This achieved by mapping scales to a unified layout, such that the musician can focus on musical expression and is relieved from the burden of choosing an appropriate scale. Our system is implemented as a dynamic MIDI scale mapper that allows to automatically select an appropriate scale based on a given chord progression. Scales may be manually pre-selected or altered as desired and scale changes may either be performed manually or automatically based on MIDI clock signals. The user interface is based on a comprehensive music-theoretic representation, which is accessible to a wide audience and does not require intense musical training.

We are planning to extend the system by adding more customization features (e.g. user-seleced color schemes), representations of other instrumental layouts (e.g. for guitar), and advanced harmonic analyses that allow for more sophisticated scale selection techniques, using machine learning on transcriptions from professional jazz musicians.

Furthermore, our system will also have to be more thoroughly evaluated by presenting it to both trained jazz musicians and as well as beginners. We hope that it will be of great pedagogical value by providing an alternative way to represent music.

5. ACKNOWLEDGMENTS

This project was conducted at the *Latour Chair in Digital and Cognitive Musicology*, generously funded by Mr Claude Latour.

References

[1] Ableton. Push, 2015. `https://www.ableton.com/en/push/`.

[2] W. Apel. *The Harvard dictionary of music*. Harvard University Press, 2003.

[3] R. W. Duffin et al. *How equal temperament ruined harmony (and why you should care)*. WW Norton & Company, 2007.

[4] A. R. François, E. Chew, and D. Thurmond. Visual feedback in performer-machine interaction for musical improvisation. In *Proceedings of the 7th International Conference on New Interfaces for Musical Expression*, pages 277–280, 2007.

[5] G. Hoffman and G. Weinberg. Gesture-based human-robot jazz improvisation. In *2010 IEEE International Conference on Robotics and Automation*, pages 582–587. IEEE, 2010.

[6] M. Levine. *The Jazz Theory Book*. Sher Music, 2011.

[7] R. Linn. Linnstrument, 2010. `https://www.rogerlinndesign.com/linnstrument`.

[8] L. McCarthy. p5.js, 2015. `https://p5js.org`.

[9] B. J. McCartin. Prelude to musical geometry. *The College Mathematics Journal*, 29(5):354–370, 1998.

[10] A. J. Milne, S. A. Herff, D. Bulger, W. A. Sethares, and R. T. Dean. XronoMorph: Algorithmic generation of perfectly balanced and well-formed rhythms. In *Proceedings of the International Conference on New Interfaces for Musical Expression*, volume 16 of *2220-4806*, pages 388–393, Brisbane, Australia, 2016. Queensland Conservatorium Griffith University.

[11] Novation. Launchpad, 2009. `https://novationmusic.com/en/launch`.

[12] E. Prout. *Harmony : its theory and practice*. seventh edition, 1903.

[13] M. Rohrmeier. Towards a generative syntax of tonal harmony. *Journal of Mathematics and Music*, 5(1):35–53, Mar. 2011.

[14] M. Rohrmeier. The syntax of jazz harmony i: Diatonic tonality, phrase structure, and form. *Music Theory and Analysis*, (6), 2020.

[15] ROLI. Juce, 2019. `https://juce.com/`.

[16] A. Schoenberg. *Theory of harmony*. University of California Press, 1978.

[17] D. Temperley. The line of fifths. *Music Analysis*, 19(3):289–319, 2000.

[18] P. T. van der Torren. Striso, a compact expressive instrument based on a new isomorphic note layout. In *Proceedings of the International Conference on New Interfaces for Musical Expression*, pages 615–620, London, United Kingdom, June 2014. Goldsmiths, University of London.