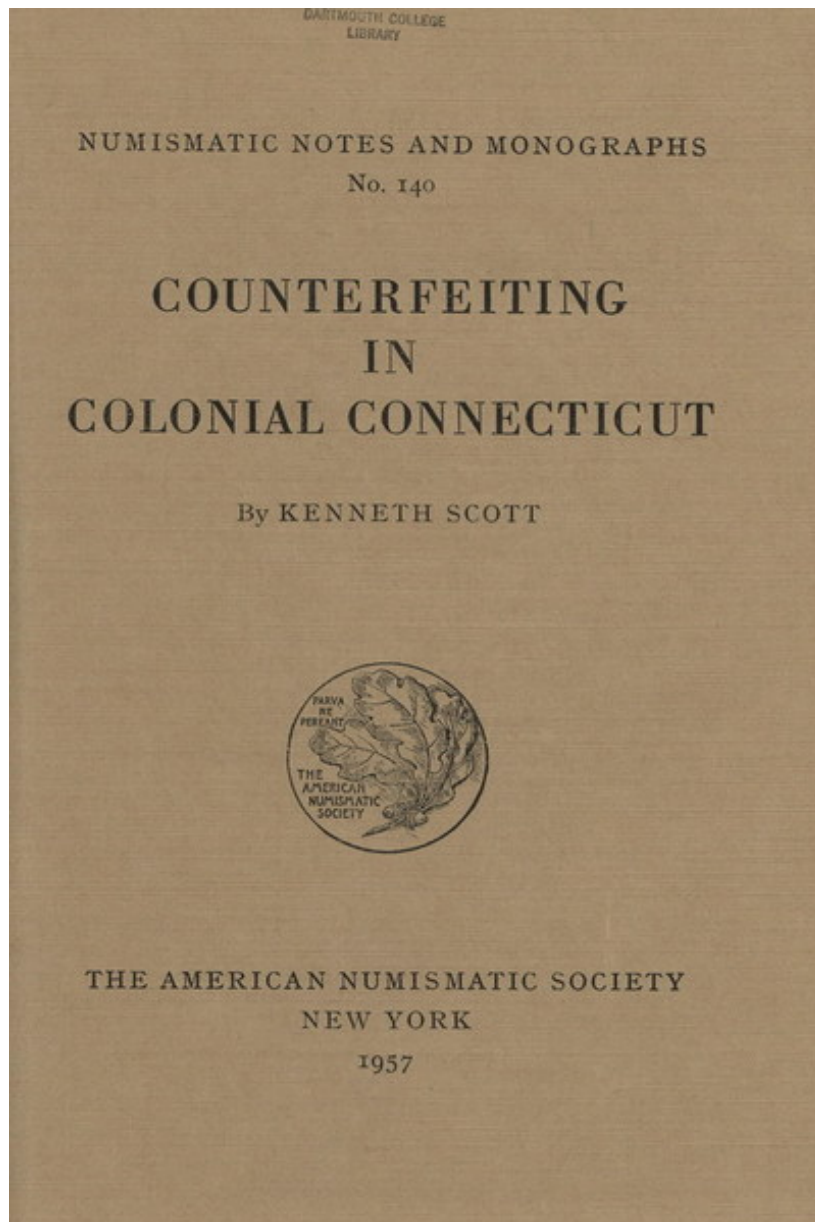


Counterfeiting in Colonial Connecticut
for guitar with low and high accompaniment and reader

dedicated to Elliot Simpson and Alex Bruck
in honor and memory of George Floyd

Preferably played in a dark or dim setting (e.g. with the least light needed by the performers) outside in the open air or any space that allows for audience members to be at least six feet apart. Proceeds generated by the piece should be donated to causes of social justice.

michael winter
(cdmx and gatlinburg, tn; 2020)



DRAFT - 2020.07.13

general remarks	1
instructions	2
musical score	6
appendix 1 - excerpts from “Counterfeiting in Colonial Connecticut” by Kenneth Scott	97
appendix 2 - SuperCollider code and Lilypond templates	103

general remarks (to optionally be used as a program note and / or read during performance)

I started writing this piece with the intention to set readings of excerpts from the book *Counterfeiting in Colonial Connecticut* written by Kenneth Scott and published by the American Numismatic Society in 1957. I was intrigued by the stories and the dry, austere nature of Scott's accounts.

My original intentions were transformed by two major crises that occurred during the development of this piece: the Covid-19 pandemic and protests sparked by the death of George Floyd, a black man brutally murdered by police. I decided to add the possibility of complementing readings from the Scott compendium with readings of texts reflecting my experience during the time in which the piece was written.

I was reluctant to connect George Floyd with counterfeiting and colonialism. Floyd was being arrested for *allegedly* using a counterfeit \$20 bill and his murder, as well as the pandemic, clearly demonstrated that inequalities accepted in colonial times have persisted. As such, the use of texts about counterfeiting in colonial America gained a whole new meaning and gravity. However, these coincidences and connections are actually quite apt. The systems enforced and perpetuated by governments today in 2020—capitalism, democracy, communism—are counterfeit. They are fraudulent implementations of ideas manipulated to satisfy greed but traded as currency for the “good” of the people; far more dangerous than the relatively benign act of passing (perhaps unknowingly) a counterfeit \$20 bill. In a more humane system, George Floyd would still be alive and a pandemic would demonstrate the resilience of our society rather than expose systemic inequalities within it.

The music of this piece was written using a coin press as a central metaphor. The underlying variables in the computer program that generates the piece vary slightly within and between each section, like variations and errors in the minting of coins.

instructions

The piece consists of a guitar part, a high accompaniment, a low accompaniment, optional electronic interludes, and readings of texts. All accompanying parts can be played by real instruments or electronically synthesized using custom software written in the SuperCollider programming language. Each of these elements are described below in more detail. The score is divided into sections and subsections. Any number of sections can be played in any order such that the piece lasts at least 10 minutes. A section may also occur multiple times. While structurally similar, each section is actually quite distinct and reordering the sections based on personal preference is encouraged.

guitar

The open strings of the guitar are tuned as follows (given by string number, a note with a deviation in cents which is 100th of a tempered semitone, and a frequency ratio from the lowest note within a set of parentheses):

VI) E down to D (1/1)

V) A +2¢ (3/2)

IV) D (2/1)

III) G down to F[♯] -14¢ (5/2)

II) B down to A -47¢ (35/12)

I) E down to C -31¢ (7/2) - Note that string II is a just 6/5 down from string I.

The notes in the guitar part of the score are written as the closest pitch in 12-tone equal-temperament *as sounds* (without cent deviations). Except in the ultimate subsection of each section, written below each note is the exact string number given as a Roman numeral and a fret number given as an Arabic superscript needed to sound the correct pitch. In the ultimate subsection of each section, the written notes all correspond to open strings. However, throughout each ultimate subsection, the guitarist can play arbitrary natural harmonics of the indicated string such that approximately half the tones are played as open strings and half are played as natural harmonics (the option of which is indicated by a diamond next to the Roman numeral below each note).

Throughout a performance of the entire piece, the guitarist should try to allow all tones to decay naturally for as long as possible beyond the written durations creating an overall resonant sound (e.g. fretting notes for as long as possible). The non-picking hand always remains in a relatively compact position on the fretboard. However, the notes will often switch between the open string and a fretted note within and around the current position. Transitions between notes on the same string can occasionally be played as hammer-ons or pull-offs even if they are not written in direct succession.

The guitar part should be present and in the foreground throughout except for each ultimate subsection, which should be played with a decrescendo corresponding to the written-in ritardando.

high accompaniment

This part can be played by any high-register, sustaining instrument. If necessary, the part can be transposed down an octave (note the *ottava* marking on the clef). Each tone should enter and exit from a soft volume or silence with a swell over the course of the tone duration such that the crescendo portion of the swell is slightly shorter than the decrescendo portion.

The part oscillates between two pitches every subsection. The higher tone (an F) is preferably played approximately 16 cents sharp (a frequency ratio of 6/5 to the next lowest D). The performer can also explore slightly altering the tuning every few tones (i.e., once a tone is altered, it should sound a few times at that exact pitch before it is altered again).

This part should be present, floating above yet not overwhelming the guitar part.

low accompaniment

This part consists of two voices that always sound together. While the noteheads of the voices are written in unison with opposing stems, the number given above indicates the difference in frequency between the two tones. This creates a beating effect caused by the slight difference with the exception of when the two voices are actually in unison (i.e. when the number given above is 0). In the second subsection of every section, the beating progressively gets slower (a smaller and smaller frequency difference between the two tones). While the indication of the frequency difference is precise, exact execution is less important than the movement towards unison. Also, the F that occurs in the penultimate subsection of every section is preferably played 16 cents sharp (a just 6/5 above the next lowest D).

The tones should have a sharp attack and a long decay such as with an electric bass that is plucked. The notation indicates this with *laissez vibrer* ties extending from the notes which are all written simply with quarter-note durations. Sustaining instruments can also be used such that each tone follows the dynamic profile described above. The part may be transposed up an octave if necessary (note the *ottava* marking on the clef).

This part should be loud and clear. The attacks should briefly overwhelm the other parts.

readings

Occasionally, texts may be read in a rather inexpressive yet clear and intelligible voice. The readings may be from accounts in the book *Counterfeiting in Colonial Connecticut* written by Kenneth Scott and published by the American Numismatic Society in 1957. A few of the accounts from the book are provided in an appendix to this document. Longer accounts may be read in part. Scott published several books about counterfeiting in colonial America. Accounts from any of these compendiums may also be read. Other texts may be considered so long as they are related to numismatics and specifically counterfeiting (e.g., excerpts from the *Lex Cornelia testamentaria nummaria* that define early counterfeiting laws in ancient Rome).

Occasionally the following texts may also be read alone or over readings from the Kenneth Scott book (via a second speaker or recording). In any order. Shorter phrases may be repeated. Portions of the “general remarks” above can also be read.

Black Lives Matter

Getting put on a ventilator was not a good sign. They were hypoxic without even realizing it. Multiple organ failure often followed. The World Health Organization finally declared a pandemic. The disease is now officially called Covid-19.

The police officer continued to kneel on his neck even though he was pleading that he could not breathe. 8 minutes and 46 seconds in total. It was so clearly murder.

We watched as country by country, they balanced or chose between health and economy. In a more humane society, this would not even be an issue. In a more humane society, the state of art and technology would be more of an indicator of well-being than gross domestic product and strictly financial measures.

The murder of George Floyd and the pandemic were inextricably linked. Not only are black people at least 5 times more likely to be in incarcerated than white people, they are also disproportionately succumbing to Covid-19.

The bears knew that the people were away and were more cavalier in their search for food outside the park. For a while, I saw bears more frequently than people.

from mark

email subject: death drop into kharachi

have you been following this?? so crazy. at first it was so strange-seeming that the crew would land with no gear, and be so baffled as to bounce three times on the engines before attempting a go around, but look at this insane approach! suddenly everything makes sense. they hadn't even grabbed the i.l.s. signal, much less got their speed, sink rate, altitude, configuration, etc under control – by the time the landing gear warning sounded (as heard in a.t.c. audio), it could have been an overlimit warning for any number of factors.

The prison-industrial complex

Thanks Paulo,

I am fine.

Very removed from everything.

With hard feelings that I am not contributing to the cause.

In solidarity with the protests.

With hopes that it signals change.

And fears that the suffering will worsen and authoritarianism will reign with an even heavier fist.

Perhaps we can connect tomorrow.

“Report fever, stiff muscles or confusion, which might mean a life threatening reaction. Or uncontrollable muscle movements may be permanent. Side effects may not appear for several weeks. Metabolic changes may occur. Movement dysfunction, restlessness, sleepiness, stomach issues are common side effects.” It is a constant cycle of political pundits acting as journalists intertwined with pharmaceutical advertisements.

Both in Brazil and in the United States, the far-right are weaponizing democratic ideals to implement authoritarianism. They want-all power. And if not, they want war. Civil unrest seems inevitable.

Not surprisingly, this was the world many corporations already wanted and envisioned. The marketing machines were essentially ready to cater to a touchless society. Corporations should be replaced by cooperations. Markets should be fair in that products are valued by the cost it takes to create them and not by manufactured desire or controlled supply.

The military-industrial complex

It is out of the scope of most peoples' vision that capitalism itself is the cause for all the suffering. As evidenced by my taxation studies, it seems clear that a proper solution to combat greed and inequality would be a wealth tax. Simply put, if everyone starts out near equal and can only accumulate wealth within their lifetime, then large wealth gaps would not persist over generations. The excess wealth could then be used for the basic needs and good of the people. There would be less incentive to ruthlessly profiteer. Poverty and inequality should not exist.

Some of the earliest laws against counterfeiting were enforced in ancient Rome as decreed in a document called the *Lex Cornelia testamentaria nummaria*. Despite legislation against counterfeiting, the Romans sometimes benefited from the practice as it inflated their currency in times of economic peril.

So many were quicker to condemn the looting than they were to condemn the murder. As Trevor Noah pointed out: it was the police that initially broke the social contract.

Complex financial instruments

Wipe down the seat with disinfectant. Mask on properly. Don't touch your face. Get through security as fast as possible once you arrive in Atlanta.

The United States constitution is not so holy. The human rights that it outlined did not apply to black people. Slavery was not even abolished until the 13th amendment was passed in 1865. And even then, there were plenty of means of oppression left intact. Protections and rights that have been painstakingly garnered over time—for and by people of color, for and by women, for and by the LGBTQ+ community—are fragile at best. Health care is a human right. Access to information is a human right. Intellectual property is the property of *all* people. Human rights are fundamental and should not be amendable.

optional interludes

Optional interludes can be inserted between sections such that they fade in starting at the ultimate subsection of each section ($x.4$), sound indefinitely, and then fade out at the beginning of the subsequent section ($(x+1).1$). This can be used to facilitate a solo performance, allowing the guitarist to stop playing and read a text during the interlude or simply to give the guitarist a rest.

The interlude is essentially a tremolo that is fed through a feedback system with a delay time that is a whole number divisor of the tremolo rate. Note that the feedback system is intentionally quite sensitive to the frequencies of the notes in the tremolo and the delay time. The tremolo should generally oscillate between two pitches with an interval between a major second and a perfect fourth apart that are centered around a pitch located between the D below middle C and an octave below. Occasionally the tremolo can focus on one of the pitches instead of oscillating between the two pitches.

This effect is modeled and implemented in the SuperCollider programming language as shown below (and also embedded in the computer program) using an oscillator as the source. However, a real instrument could be used as a source into a similar feedback system.

The interlude tremolo may also be played before the piece starts and after the piece ends.

```

1  (//note that this is sensitive to frequency and tremolo rate inputs
2  SynthDef(\interludeTremolo, {arg gate = 0, amp = 1, freq1, freq2, tremRate;
3    var tremoloTrig, trem, freq, sig, feedback, fade;
4    //fast tremolo - note that this can be slower so long as the delaytime of the feedback remains short
5    tremoloTrig = Impulse.kr(tremRate);
6    //tremolo between two notes
7    trem = Select.kr(Stepper.kr(tremoloTrig, 0, 0, 1), [freq1, freq2]);
8    //occasionally tremolo on same note
9    freq = Select.kr(TWChoose.kr(Dust.kr(10), [0, 1, 2], [5, 1, 1], 1), [trem, freq1, freq2]);
10   //generate signal
11   sig = VarSaw.ar(freq, 0, 0.3, 0.1) * EnvGen.kr(Env.perc(0.01, 0.1), tremoloTrig);
12   //feedback
13   feedback = CombC.ar(sig, 0.2, tremRate.reciprocal, 5);
14   fade = feedback * EnvGen.kr(Env.asr(15, 1, 15, \sine), gate) * amp * 0.75;
15   Out.ar([0, 1], fade);
16 }) .add;
17 )
18 (//example usage
19 var center, interval, freq1, freq2, tremRate;
20 center = 50 - 12.0.rand;
21 interval = 3.0.rand + 2;
22 freq1 = center + (interval / 2);
23 freq2 = center - (interval / 2);
24 tremRate = 50 + 4.0.rand2;
25 t = Synth(\interludeTremolo, [\gate, 1, \amp, 1, \freq1, freq1, \freq2, freq2, \tremRate, tremRate])
26 )

```

interlude synth code in SuperCollider

SuperCollider program

While the piece has been written such that it can be played without the aid of a computer, a custom program written in the SuperCollider language can be used to synthesize any of the accompanying parts. The program also synthesizes the guitar part using a Karplus-Strong plucked string model, but this should only be used for auditioning and practice. The high accompaniment part is synthesized using sine tones with skewed, bell-shaped envelopes and the low accompaniment is synthesized using sine tones and envelopes that approximate a plucked electric bass.

The application source code is appended at the end of this score and downloadable from a git repository at:

https://gitea.unboundedpress.org/mwinter/counterfeiting_in_colonial_connecticut

The application provides a transport window to control playback and set variables as well as a basic mixing console to control the levels of the various sonic elements of the piece. The program also allows new versions of the piece to be generated and transcribed. Note that most of the code facilitates usability, playback, and transcription. However, the music of the piece is completely generated by the algorithm in `cicc_musical_data_generator.scd`. A help / readme file is included with the application documenting its functionality and use. To launch the application, execute `cicc_main.scd` in SuperCollider (on Linux, this is achieved by pressing `cmd+enter` with the cursor anywhere within the code block).

The generation of this document (using LaTeX) contains a version date at the bottom of this page in order to help track changes and the git repository will also detail commit changes. The piece was written using SuperCollider version 3.11.0 and Lilypond version 2.18.83.



application user interface

version generated: 2020.07.13

I would like to extend a special thanks to Alex Bruck, Freeman Edwards, Paulo Dantas, Lauren Pratt, and Elliot Simpson. I first encountered the Kenneth Scott compendium at Alex's apartment months prior to leaving Mexico because of the pandemic. It was also around that time that I first communicated with Freeman after we discovered each others' work. However, our friendship really started to grow throughout the pandemic and his comments and suggestions regarding the score were extremely helpful. Similarly, Paulo and Lauren, with whom I communicated frequently while in quarantine, made several suggestions that greatly improved the general remarks. Finally, Elliot's advice was integral. He generously answered questions about notation that led to the final format of the musical score for the piece, which is written for him as a performer in mind.

seed: 20200525

michael winter
(cdmx and gatlinburg, tennessee; 2020)

DRAFT - 2020.07.13

21 (8)

high

guitar

low

II⁰ I⁰ IV⁵ VI⁰ IV⁰ III⁷ III⁶ IV⁰ III⁰ VI⁰ IV⁰ III⁵

25 (8)

high

guitar

low

VI⁰ IV⁰ III⁰ II⁰ IV⁰ III⁵ VI⁵ III⁰ VI⁵ VI⁵ VI⁵ IV⁵ VI⁰ I⁸ I⁶ II⁰ IV⁰ VI⁰

29 (8)

high

guitar

low

I⁰ II⁵ IV³ III⁵ VI⁵ II⁰ IV⁵ I⁸ I⁶ II⁰ IV⁰

1.2

7.0

33 (8)

high

guitar

low

VI⁰ I⁰ II⁵ IV³ III⁵ VI⁵ II⁰ IV⁵

6.5

37 (8)

high

guitar

low

I⁸ I⁶ II⁰ IV⁰ VI⁰ I⁰ II⁵ IV³ III⁵

5.9

41 (8)

high

guitar

low

VI⁵ II⁰ IV⁵ VI⁰ I⁶ IV⁰ I⁰ II⁵ IV³

45 (8)

high

guitar

low

5.4

III⁵ VI⁵ II⁰ VI⁰ I⁶ II⁰ I⁰ II⁵

49 (8)

high

guitar

low

4.8

III⁵ VI⁵ II⁰ IV⁵ I⁸ I⁶ II⁰ II⁵ IV³ VI⁰

53 (8)

high

guitar

low

4.3

III⁵ VI⁵ II⁰ IV⁵ VI⁰ I⁸ I⁶ II⁰ IV⁰

57 (8)

high

guitar

low

3.8

I⁰ II⁵ III⁵ VI⁵ IV⁵ I⁸ I⁶ II⁰ IV⁰ VI⁰

61 (8)

high

guitar

low

3.2

II⁵ IV³ III⁵ VI⁵ II⁰ IV⁵ I⁸ VI⁰

65 (8)

high

guitar

low

2.7

I⁶ II⁰ IV⁰ VI⁰ I⁰ II⁵ IV³ III⁵ VI⁵

69 (8)

high

guitar

low

IV⁵ I⁸ I⁶ II⁰ IV⁰ VI⁰ II⁵ IV³

73 (8)

high

guitar

low

2.2

III⁵ VI⁵ II⁰ IV⁵ I⁸ II⁰ IV⁰ VI⁰

77 (8)

high

guitar

low

1.6

I⁰ II⁵ III⁵ VI⁵ II⁰ IV⁵ II⁰ VI⁰

81 (8)

high

guitar

low

IV⁰ I⁰ IV³ III⁶ VI⁶ IV⁶

1.1

85 (8)

high

guitar

low

VI⁰ I⁶ I⁶ IV⁰ I⁰ II⁶ IV³ III⁶ VI⁶

0.5

89 (8)

high

guitar

low

II⁰ IV⁶ VI⁰ I⁸ I⁶ II⁰ I⁰ II⁶ IV³

93 (8)

high

guitar

low

1.3

VI³ I⁴ II⁰ I⁰ II⁰ I⁰ II⁵ II³ I⁰ VI⁰ V⁶ V⁶ VI⁰ V⁰ VI² V⁰ VI⁰ V⁴ VI⁰ V⁰

0.0

97 (8)

high

guitar

low

VI⁰ V⁰ VI⁰ V⁰ VI⁰ V⁰ VI⁰ V³ III⁵ III⁴ I⁰ V³ V¹ IV³ II³ II¹ II⁰ II⁰ VI⁰ VI⁰ VI⁰

101 (8)

high

guitar

low

3.2

IV³ II⁰ III⁰ IV⁰ II⁰ III⁴ VI⁰ II⁰ IV² II⁰ IV⁰ I² IV⁰ I¹ IV⁰ V⁰ III⁰ IV⁰ III⁰

105 (8)

high

guitar

low

2.1

IV⁰ VI⁰ IV⁰ VI⁰ IV⁰ I¹ III⁰ II⁰ IV⁰ V⁰ III³ IV⁰ II⁰ IV⁰ II⁰ VI⁰ IV⁰ II⁰ VI⁰ I⁰ IV⁰ VI⁰

109 (8)

high

guitar

low

0.0

II⁰ V⁰ III³ III¹ IV⁰ VI⁰ I⁰ II⁰ VI⁰ III³ I⁰ IV⁰ II⁰ VI⁰

112 (8)

high

guitar

low

IV⁰ II⁰ VI⁰ I⁰ VI⁰ III⁰

116 (8)

high

guitar

low

VI⁰ III⁰ VI⁰ III⁰

(120) (8)

high

guitar

low

II°

I°

V°

(124) (8)

high

guitar

low

III°

VI°

(128) (8)

high

guitar

low

2.1
2/2

high

guitar

low

0.0

(8)

II¹² I¹⁰ I⁹ V⁰ III⁰ VI⁰ II¹⁰ I⁰ V⁰ III⁹ I⁰ III⁹ II⁰ IV⁹ VI⁰ IV⁰ II⁹

high

guitar

low

(8)

IV⁹ II⁹ VI⁰ IV⁰ II⁹ VI¹² VI¹⁰ I⁰ II⁸ I⁸ II⁷ III⁰ V⁸ I⁰ III⁹ III⁸ V⁰ IV⁰

high

guitar

low

(8)

II⁰ I⁷ IV⁷ V⁷ II⁰ VI⁰ I⁷ III⁰ I⁷ IV⁰ VI¹⁰ VI⁹ III⁰ I⁷ I⁶

2.2
2/2

high

guitar

low

7.0

(8)

II⁰ I⁷ IV⁷ V⁷ II⁰ I⁷ I⁷ IV⁰ VI⁹ III⁰

high

guitar

low

(8)

I⁷ I⁶ II⁰ I⁷ V⁷ II⁰ I⁷

5.6

high

guitar

low

(8)

(8)

(8)

4.2

VI⁰ III⁰ I⁷ VI¹⁰ VI⁹ III⁰ I⁷ II⁰ I⁷ IV⁷ V⁷

high

guitar

low

(8)

(8)

(8)

2.8

II⁰ VI⁰ III⁰ IV⁰ VI¹⁰ VI⁹ III⁰ I⁶ II⁰ I⁷

high

guitar

low

(8)

(8)

(8)

IV⁷ V⁷ I⁷ III⁰ I⁷ IV⁰ VI¹⁰ VI⁹ III⁰

high

guitar

low

(8)

(8)

(8)

1.4

I⁶ II⁰ I⁷ IV⁷ V⁷ VI⁰ I⁷ VI⁰

high

guitar

low

(8)

(8)

(8)

0.0

2.3

I⁷ IV⁰ IV⁰ III⁸ III⁷ I⁶ IV⁰ I⁶ IV⁰ V⁶ VI⁰ I⁶ IV⁰ V⁰ VI⁹ VI⁷ I⁰

high

guitar

low

(40) (8)

IV⁷ V⁰ VI⁷ VI⁵ I⁰ IV⁰ V⁴ VI⁵ VI⁴ I⁰ IV⁰ V⁴ VI⁰ I⁶ I⁴ IV⁰ V⁰ VI⁴ I⁴ IV⁰ V⁰ VI⁰ I⁴

high

guitar

low

(44) (8)

IV⁶ V⁰ VI⁰ I⁴ IV⁵ V⁰ VI⁰ V⁰ VI⁰ I⁴ IV⁵ IV⁴ III⁶ IV⁰ II⁰ III⁰ VI⁴

high

guitar

low

(48) (8)

IV⁴ II⁰ III⁰ IV⁰ VI⁰ IV⁴ I⁰ VI⁴ III⁰ II⁰ III⁰ I⁴ III⁰ V⁰ I⁰ VI⁴ II⁰ IV⁴ I⁰

high

guitar

low

(52) (8)

III⁴ IV⁰ III⁰ IV⁴ VI⁰ I⁰ III⁰ II⁵ III⁰ I⁰ V⁰ IV⁴ II⁰ III⁴ III³ I⁰ V⁰ IV⁴ IV²

high

guitar

low

(56) (8)

II⁰ III⁰ I⁴ I² V⁰ IV² V⁰ IV² III⁰ II⁰ VI⁴ VI² V⁰ IV⁰ III³ II⁰ I² V² IV⁰ III⁰

high

guitar

low

(60) (8)

IV⁰ III⁰ I⁰ IV⁰ VI⁰ VI² III⁰ I⁰ IV⁰ III⁰ I⁰ II⁴ VI² IV⁰ III⁰ I⁰ II⁰ IV⁰ III³ III¹

4.1

high

guitar

low

(64) (8)

I⁰ II⁰ VI⁰ IV⁰ III¹ I⁰ II⁰ VI⁰ IV⁰ III⁰ I⁰ II⁴ III³ VI¹ IV⁰ III⁰ I⁰ II⁰ VI⁰ III⁰ I⁰

2.8

high

guitar

low

(68) (8)

II⁰ III⁰ I¹ II⁰ III⁰ I¹ I⁰ II⁰ III⁰ I⁰ II³ II² III⁰ I⁰ II² II⁰ III⁰ I⁰ III⁰ I⁰ II⁰

high

guitar

low

(72) (8)

2.4

V⁰ IV⁰ III⁰ I⁰ II⁰ V⁰ IV⁰ III⁰ I⁰ II⁰ V⁰ IV⁰ I⁰

0.0

high

guitar

low

(76) (8)

IV⁰ VI⁰ V⁰ II⁰ III⁰

3.1

high

guitar

low

0.0

5

high

guitar

low

9

high

guitar

low

13

high

guitar

low

17

high

guitar

low

Chord symbols and musical notation are present throughout the system.

high

guitar

low

(21) (8)

VI⁰ III¹⁰ VI⁰ III⁰ VI⁷ III⁰ VI⁷ III⁹ VI⁰ III⁶ VI⁶ IV⁷ V⁵ VI⁰ IV⁰ V⁰ VI⁰ III⁶ V⁵ I⁷

high

guitar

low

(25) (8)

3.2

6.0

IV⁶ VI⁶ VI⁵ III⁷ V⁰ I⁰ IV⁰ VI⁰ V⁰ I⁶ IV⁰ VI⁰ III⁷ III⁵ V⁴ I⁶ IV⁰ VI⁰

high

guitar

low

(29) (8)

5.5

III⁷ III⁵ V⁴ I⁶ IV⁰ VI⁰ III⁷ III⁵ V⁴

high

guitar

low

(33) (8)

5.0

4.5

I⁶ VI⁰ III⁷ III⁵ I⁶ IV⁰ VI⁰ III⁷ V⁴

high

guitar

low

(37) (8)

4.0

3.5

3.0

I⁶ VI⁰ III⁵ V⁴ I⁶ IV⁰ VI⁰ III⁷ III⁵ V⁴ I⁶

high (41) (8)

guitar (8)

low (8)

IV⁰ VI⁰ V⁴ I⁶ VI⁰ III⁷ III⁵ I⁶ IV⁰

2.5 2.0

high (45) (8)

guitar (8)

low (8)

VI⁰ III⁷ III⁵ I⁶ VI⁰ V⁴ I⁶ IV⁰

1.5 1.0

high (49) (8)

guitar (8)

low (8)

III⁷ V⁴ IV⁰ VI⁰ III⁷ III⁵ I⁶ IV⁰ III⁰ VI⁰

0.5 0.0

3.3

high (53) (8)

guitar (8)

low (8)

V⁰ I⁰ IV⁰ VI⁰ IV⁵ II⁰ V⁰ VI⁴ IV⁰ II⁰ V⁰ VI⁰ IV⁰ II⁰ #V⁴ II⁷

high (57) (8)

guitar (8)

low (8)

II⁶ V⁰ IV⁰ I⁵ #V⁴ V³ II⁰ V⁰ II⁰ II⁰ IV⁵ IV³ III⁴ V⁰ VI⁰ II⁴ IV⁰ III⁰ V⁰

61 (8)

high

guitar

low

III² II³ V⁰ VI³ I⁰ IV³ IV² III⁰ V⁰ VI⁰ I⁴ IV⁰ III⁰ V⁰ VI⁰ I⁰ IV⁰ V¹ III⁰ I³

65 (8)

high

guitar

low

IV⁰ V⁰ III⁰ I³ IV⁰ V⁰ III⁰ VI¹ I⁰ IV⁰ V⁰ III⁰ VI⁰ IV⁰ VI⁰ V⁰ III⁰

3.9

69 (8)

high

guitar

low

I³ I⁰ I³ I¹ I⁰ IV⁰ II⁰ IV⁰ II⁰ III⁰ IV⁰ V⁰ VI⁰ II⁰

1.1

73 (8)

high

guitar

low

II⁰ III⁰ I⁰ VI⁰ V⁰ II² I⁰ II² II¹ I⁰ V⁰ VI⁰

3.4

76 (8)

high

guitar

low

II⁰ I⁰ V⁰ VI⁰ I⁰ V⁰ II⁰

0.0

high

guitar

low

VI°

V°

The musical score for "The Sound of Silence" is presented in three staves. The top staff, labeled "high", is in treble clef and contains a melodic line with a key signature of one flat (Bb) and a time signature of 4/4. It begins with a whole rest, followed by a half note Bb, a half note A, and a half note G. The middle staff, labeled "guitar", is in treble clef and contains a melodic line with a key signature of one flat (Bb) and a time signature of 4/4. It begins with a whole rest, followed by a half note Bb, a half note A, and a half note G. The bottom staff, labeled "low", is in bass clef and contains a melodic line with a key signature of one flat (Bb) and a time signature of 4/4. It begins with a whole rest, followed by a half note Bb, a half note A, and a half note G. The score is marked with a key signature of one flat (Bb) and a time signature of 4/4. The tempo is marked "Andante". The score is marked with a key signature of one flat (Bb) and a time signature of 4/4. The tempo is marked "Andante".

4.1

high

guitar

low

0.0

5

high

guitar

low

9

high

guitar

low

13

high

guitar

low

17

4.2

high

guitar

low

6.0

high (21) (8)

guitar (8) Π^6 Π^6 VI^7 Π^0 Π^6 Π^5 IV^6 Π^6 V^0 IV^6 Π^0

low (8) 5.6

high (25) (8)

guitar (8) IV^0 Π^6 V^0 IV^0 Π^6 VI^0 Π^6 VI^7 Π^0

low (8)

high (29) (8)

guitar (8) Π^5 IV^6 Π^6 V^0 IV^6 Π^0 V^6 IV^0 Π^6

low (8) 5.1

high (33) (8)

guitar (8) V^0 IV^0 Π^6 VI^0 Π^6 VI^7 Π^0 VI^0 IV^6 Π^6 V^0

low (8) 4.7

high (37) (8)

guitar (8) IV^6 IV^0 Π^6 V^0 IV^0 Π^6 VI^0 VI^7 Π^0 VI^0 Π^6

low (8)

41 (8)

high

guitar

low

4.3

Chord symbols: Π^5 , V^0 , Π^0 , V^6 , Π^6 , IV^0 , Π^6

45 (8)

high

guitar

low

3.9

Chord symbols: VI^0 , Π^6 , VI^7 , Π^0 , VI^0 , Π^5 , IV^6 , Π^6

49 (8)

high

guitar

low

Chord symbols: V^0 , IV^6 , Π^0 , IV^0 , Π^6 , V^0 , IV^0 , VI^0 , VI^7

53 (8)

high

guitar

low

3.4

Chord symbols: Π^0 , VI^0 , Π^5 , IV^6 , IV^6 , Π^0 , V^5 , IV^0 , Π^6 , V^0

57 (8)

high

guitar

low

Chord symbols: IV^0 , Π^6 , VI^0 , Π^6 , Π^0 , VI^0 , Π^6 , Π^5

high

guitar

low

3.0

(s)

(s)

(s)

IV⁶ II⁶ V⁰ IV⁶ II⁰ IV⁰ II⁶ V⁰ IV⁰

high

guitar

low

(s)

(s)

(s)

II⁶ VI⁰ II⁶ VI⁷ II⁰ II⁵

high

guitar

low

2.6

(s)

(s)

(s)

II⁶ V⁰ V⁶ IV⁰ II⁶ V⁰ IV⁰ II⁶

high

guitar

low

2.1

(s)

(s)

(s)

VI⁰ VI⁷ VI⁰ II⁶ II⁶ IV⁶ V⁰

high

guitar

low

(s)

(s)

(s)

IV⁶ II⁰ V⁶ IV⁰ II⁶ IV⁰ II⁶

81 (8)

high

guitar

low

VI⁰ II⁶ VI⁷ II⁰ VI⁰ II⁶ IV⁶ V⁰

1.7

85 (8)

high

guitar

low

IV⁶ II⁰ V⁶ II⁶ V⁰ IV⁰ II⁶

89 (8)

high

guitar

low

VI⁰ II⁶ VI⁷ II⁰ VI⁰ II⁶ II⁶

93 (8)

high

guitar

low

IV⁶ IV⁶ II⁰ V⁶ IV⁰ II⁶ V⁰ IV⁰ II⁶

1.3

97 (8)

high

guitar

low

VI⁰ II⁶ VI⁷ II⁰ VI⁰ II⁶ IV⁶

0.9

high (101) (8)

guitar (8)

low (8)

II⁶ V⁰ IV⁶ II⁰ V⁶ IV⁰ II⁶

high (105) (8)

guitar (8)

low (8)

IV⁰ II⁶ VI⁰ II⁶ VI⁷ VI⁰ II⁶

high (109) (8)

guitar (8)

low (8)

II⁵ IV⁶ II⁶ V⁰ IV⁶ II⁰

0.4

high (113) (8)

guitar (8)

low (8)

IV⁰ II⁶ V⁰ IV⁰ II⁶ VI⁰ II⁶ VI⁷

high (117) (8)

guitar (8)

low (8)

II⁰ II⁵ VI⁶ II⁵ II⁴ II⁴ VI⁰ VI⁶ II⁰ II⁴ VI⁰ VI⁰ III⁰ VI⁶ VI⁵

4.3

0.0

121 (8)

high

guitar

low

Chords: III^5 , VI^0 , III^0 , VI^0 , I^0 , VI^0 , II^0 , V^4 , IV^0 , I^0 , VI^4 , II^0 , V^0 , IV^0 , VI^4 , I^0 , III^0 , II^0 , VI^4 , I^6 , III^0 , II^0 , VI^0

125 (8)

high

guitar

low

Chords: I^0 , III^0 , II^4 , II^3 , VI^0 , I^0 , III^4 , II^0 , I^5 , III^0 , II^0 , VI^4 , VI^3 , III^4 , II^0 , I^0 , V^3 , II^3 , IV^0 , VI^3 , V^0

129 (8)

high

guitar

low

Chords: II^0 , IV^4 , VI^0 , I^6 , I^4 , V^0 , II^2 , IV^2 , V^0 , VI^0 , II^1 , IV^0 , V^0 , VI^2 , IV^2 , IV^1 , III^0 , I^0 , IV^0

133 (8)

high

guitar

low

Chords: V^0 , II^0 , III^3 , I^2 , IV^0 , V^3 , V^2 , II^0 , III^1 , I^0 , IV^0 , V^0 , II^0 , III^0 , I^0 , IV^0 , V^0 , II^0 , I^0 , VI^0

4.5

137 (8)

high

guitar

low

Chords: II^0 , IV^0 , I^0 , VI^0 , II^0 , IV^0 , V^0 , VI^0 , V^0 , VI^0 , II^0 , I^0 , IV^0 , V^0 , VI^0 , III^0 , II^0 , I^0 , III^0 , II^0 , I^0

2.0

high

guitar

low

(144) (8)

(8)

I° VI° IV° I° VI° IV° VI°

The musical score is for the song "The Sound of Silence" by Simon & Garfunkel. It is written in 3/4 time. The score consists of three staves: "high" (treble clef), "guitar" (treble clef), and "low" (bass clef). The "high" staff begins with a circled measure number 148 and contains a half note G5. The "guitar" staff contains a half note G4, followed by a half note F#4, and then a half note E4. The "low" staff contains a half note G2. The guitar part includes a capo symbol (IV) and a key signature change to one sharp (F#) indicated by a diamond symbol.

high

guitar

low

(8)

(8)

(8)

VI⁹ III⁷ III⁶ V⁰ II⁰ IV⁶ III⁶ V⁷ II⁰ IV⁶ VI⁷ III⁰ IV⁵ II⁷ I⁰ VI⁷ VI⁶ IV⁵

5.2

2/2

20 (8)

high

guitar

low

23 (8)

high

guitar

low

27 (8)

high

guitar

low

31 (8)

high

guitar

low

35 (8)

high

guitar

low

7.0

6.0

5.0

4.0

II⁰ I⁷ II⁶ V⁵ IV⁰ II⁶ II⁴ II⁰ IV⁶ VI⁷ III⁰ VI⁰ IV⁵

II⁷ I⁰ VI⁷ VI⁶ IV⁵ II⁰ I⁷ II⁶ II⁶ II⁰ IV⁶ VI⁷

III⁰ IV⁵ II⁷ I⁰ VI⁷ VI⁶ IV⁵ II⁰ I⁷ II⁶

V⁵ IV⁰ II⁶ II⁴ II⁰ IV⁶ VI⁷ III⁰ IV⁵ VI⁰

II⁷ I⁰ VI⁷ VI⁶ IV⁵ II⁰ I⁷ II⁶ IV⁰ II⁶ II⁴ II⁰

high

guitar

low

(8)

39

IV⁶ VI⁷ III⁰ VI⁰ II⁷ I⁰ VI⁷ VI⁶ IV⁵ II⁰ I⁷ II⁶

high

guitar

low

(8)

43

V⁵ IV⁰ II⁶ II⁴ II⁰ IV⁶ VI⁷ III⁰

3.0

high

guitar

low

(8)

47

IV⁵ II⁷ I⁰ VI⁷ VI⁶ IV⁵ II⁰ II⁶ V⁵ IV⁰ II⁶ II⁰

2.0

high

guitar

low

(8)

51

IV⁶ III⁰ VI⁰ IV⁵ II⁷ I⁰ VI⁷ VI⁶ IV⁵ II⁰ I⁷

high

guitar

low

(8)

55

V⁵ IV⁰ II⁴ IV⁶ IV⁵ II⁷ VI⁷ VI⁶ IV⁵ II⁰ I⁷

1.0

59 (8) 5.3

high

guitar

low

(8)

V⁵ IV⁰ II⁶ II⁴ V⁴ IV⁰ I⁰ V⁰ I⁷ V⁴ IV⁴ V⁰ II⁰ IV⁴ V⁴

0.0

63 (8)

high

guitar

low

(8)

II⁴ IV⁰ V⁰ II⁰ IV⁴ V⁴ II⁴ III⁴ V⁰ VI⁰ III⁰ V⁰ VI⁰ V⁰ VI⁶ VI⁴ V⁴ VI⁴ II⁰

67 (8)

high

guitar

low

(8)

VI⁰ V⁴ II⁰ VI⁰ V⁰ II⁴ VI⁰ V⁴ IV⁴ V⁴ IV⁰ V⁴ III⁴ V⁴ VI⁴ I⁰ VI⁴ V⁰ II⁰ VI⁴ II⁴ VI⁰

71 (8) 3.1

high

guitar

low

(8)

I⁷ I⁵ VI⁴ VI² V⁴ V³ II⁴ III⁰ I⁰ VI⁰ V³ V² II⁰ III⁰ I⁶ I⁴ VI⁰ V⁰ II³ III⁰ VI⁰ IV⁰ II⁰

75 (8) 2.6

high

guitar

low

(8)

III⁴ IV³ II³ II² III⁰ VI⁰ IV³ IV¹ II¹ III⁴ III³ VI⁰ IV¹ II¹ III⁰ IV⁰ I⁰ V⁰ II⁰ III² IV⁰

79 (8)

high

guitar

low

5.4

3

2

0.0

Chord symbols: I^4 , I^2 , V^0 , II^0 , III^1 , IV^0 , I^1 , V^0 , II^0 , III^0 , IV^0 , I° , V° , II° , III°

82 (8)

high

guitar

low

Chord symbols: IV° , I° , V° , II° , III° , IV° , I° , III° , II° , IV°

86 (8)

high

guitar

low

Chord symbols: I° , III° , II°

[illegible]

21 (8)

high

guitar

low

Chords: III^0 , VI^0 , I^0 , II^6 , III^6 , VI^0 , I^0 , II^0 , III^0 , VI^8 , I^0 , II^8 , III^0 , VI^8 , I^0 , II^6 , I^0 , II^0 , V^0

25 (8)

high

guitar

low

Chords: I^6 , II^0 , I^8 , II^0 , V^9 , I^6 , II^0 , V^8 , I^0 , II^0 , V^0 , I^5 , II^0 , V^0 , IV^6 , VI^0 , V^0 , II^0 , III^0 , IV^0 , I^0 , V^6 , II^0

29 (8)

high

guitar

low

Chords: V^5 , I^5 , II^0 , V^5 , I^5 , II^0 , V^5 , I^5 , II^0 , V^5 , I^5 , II^0 , V^5 , III^8 , III^6 , I^5 , III^5 , IV^0 , V^5 , III^6

33 (8)

high

guitar

low

Chords: IV^5 , V^0 , II^0 , VI^0 , III^0 , IV^0 , V^5 , II^7 , VI^8 , III^0 , IV^0 , V^0 , II^0 , VI^0 , I^0 , IV^5 , II^0 , I^0

37 (8)

high

guitar

low

Chords: IV^0 , II^7 , I^0 , IV^0 , II^7 , I^0 , IV^0 , II^0 , I^0 , IV^0 , II^5 , I^0 , IV^0 , II^0 , I^0 , IV^5 , II^0 , I^0 , IV^0 , II^5 , I^0 , IV^0 , II^0

41 (8)

high

guitar

low

VI⁰ IV⁰ V⁵ I⁶ III⁵ I⁰ II⁰ IV⁰ I⁰ II⁰ III⁵ V⁰ III⁰ I⁰ III⁰

45 (8)

high

guitar

low

I⁰ III⁵ I⁶ II⁰ VI⁰ VI⁶ V⁵ III⁰ IV⁰ II⁶ IV⁰ VI⁷ IV⁰ VI⁷ II⁵ II⁴ IV⁰ II⁰ IV⁵ IV⁴ V⁰

49 (8)

high

guitar

low

II⁰ I⁰ VI⁶ II⁰ VI⁵ II⁰ I⁰ V⁰ IV⁴ V⁰ I³ II⁵ II⁴ II⁰

6.2

6.0

52 (8)

high

guitar

low

V⁰ II⁰ I⁰ VI⁶ II⁰ VI⁵ II⁰ I⁰ V⁰ IV⁴

56 (8)

high

guitar

low

V⁰ I³ II⁶ IV⁰ II⁰ IV⁴ V⁰ II⁰ I⁰ VI⁶ II⁰

5.0

high

guitar

low

(60) (8)

VI⁵ II⁰ V⁰ IV⁴ V⁰ I³ II⁴ IV⁰ IV⁵ IV⁴

4.0

high

guitar

low

(64) (8)

V⁰ II⁰ I⁰ VI⁶ II⁰ VI⁵ II⁰ I⁰ IV⁴ V⁰

high

guitar

low

(68) (8)

I³ II⁵ II⁴ IV⁰ II⁰ IV⁵ V⁰ I⁰ VI⁶ II⁰

3.0

high

guitar

low

(72) (8)

VI⁵ I⁰ V⁰ IV⁴ V⁰ I³ II⁵ II⁴

2.0

high

guitar

low

(76) (8)

IV⁰ II⁰ IV⁵ V⁰ II⁰ I⁰ VI⁶ VI⁵ I⁰ V⁰

80 (8)

high

guitar

low

IV⁴ V⁰ I³ II⁵ IV⁰ II⁰ IV⁴ V⁰ II⁰ I⁰ #VI⁶ II⁰

1.0

84 (8)

high

guitar

low

VI⁵ II⁰ I⁰ V⁰ V⁰ I³ V⁰ I³ III⁴ V⁰ VI⁰

6.3

0.0

87 (8)

high

guitar

low

III⁴ III³ III⁰ III³ #VI⁴ III² VI² VI⁰ IV² III² I⁰ VI⁰ IV⁰ III⁰ V⁰ I⁰ VI⁰

91 (8)

high

guitar

low

III⁰ V³ I³ I¹ VI⁰ III¹ V¹ VI⁰ III⁰ V¹ I⁰ VI⁰ III⁰ V⁰ IV⁰ I⁰ V⁰ IV¹

3.7

95 (8)

high

guitar

low

I⁰ IV⁰ I⁰ III⁰ II⁰ V⁰ I⁰ III⁰ II² V⁰ I⁰ III⁰ II⁰ V⁰ I⁰ III⁰ II⁰ V⁰ II⁰ I⁰ IV⁰ V⁰ III⁰

1.1

99 (8) 6.4

high

guitar

low

VI⁰ II¹ V⁰ III⁰ II⁰ V⁰ III⁰ II⁰ V⁰ III⁰ II⁰ III⁰ V⁰

0.0

103 (8)

high

guitar

low

II⁰ VI⁰ I⁰ III⁰ V⁰

107 (8)

high

guitar

low

II⁰ VI⁰ I⁰

111 (8)

high

guitar

low

7.1

high

guitar

low

0.0

5

high

guitar

low

9

high

guitar

low

13

high

guitar

low

17

high

guitar

low

Chord symbols and fingerings are provided for the guitar part in each system.

high

guitar

low

(21) (8)

III⁶ I⁶ V⁶ VI⁶ IV⁰ II⁰ I⁰ VI⁰ V⁶ III⁶ II⁰ I⁰ II⁷ IV⁰ I⁶ VI⁶ V⁰ I⁶ V⁶

high

guitar

low

(25) (8)

I⁰ V⁰ I⁶ V⁰ VI⁰ II⁰ IV⁰ V⁰ IV⁹ III⁰ VI⁰ IV⁰ III⁶

high

guitar

low

(28) (8)

7.2
2
2

VI⁰ IV⁰ III⁰ VI⁶ IV⁷ III⁴ IV⁰ III⁰ VI⁶ IV⁷ IV⁰ III⁰ VI⁶

5.0 4.2

high

guitar

low

(31) (8)

IV⁷ III⁴ IV⁰ III⁰ VI⁶ IV⁷ III⁴

3.3

high

guitar

low

(35) (8)

IV⁰ III⁰ VI⁶ IV⁷ III⁴ IV⁰ III⁰ VI⁶ IV⁷ III⁴

2.5 1.7

7.3

high

guitar

low

IV⁰ VI⁶ IV⁷ III⁴ VI⁰ V⁰ II⁰ IV⁰ VI⁰

0.8 0.0

high

guitar

low

III⁴ II⁷ II⁶ III⁴ I⁰ II⁰ III⁰ I⁶ I⁴ II⁰ III⁰ I⁴ IV⁷ IV⁵ II⁵ V⁴ III⁰ IV⁰ III⁰ IV⁴

high

guitar

low

VI⁵ V⁰ I⁰ II⁰ IV⁰ VI⁰ V² I⁰ II⁵ I⁴ I² III² I² III² I⁰ II⁵ II⁴ V²

3.3

high

guitar

low

IV⁴ IV³ III⁰ IV³ IV² V⁰ I⁰ III⁰ IV⁰ I² IV⁰ II⁰ V⁰ VI⁵ VI³ III⁰ I¹ V⁰ II⁴

2.0

high

guitar

low

II² VI² I⁰ III⁰ I⁰ III⁰ VI⁰ VI² VI⁰ I⁰ III⁰ II⁰ I⁰ II⁰ I⁰ II⁰

0.0

high (58) (8)

guitar (8)

low (8)

I° II° I° IV° V° III° II°

high (62) (8)

guitar (8)

low (8)

IV° V° III°

high (66) (8)

guitar (8)

low (8)

II°

8.1

high

guitar

low

0.0

5

high

guitar

low

9

high

guitar

low

13

high

guitar

low

17

high

guitar

low

Chord symbols and musical notation are present throughout the system.

8.2
2

3

high

guitar

low

(s)

VI⁰ V⁰ I⁰ II⁶ VI⁰ III⁰ VI⁰ IV⁰ III⁶ VI⁴ VI⁰ 7.0 VI⁰

high

guitar

low

(s)

IV⁰ III⁶ VI⁴ VI⁰ 6.2 III⁰ VI⁰ IV⁰ III⁶ VI⁴

high

guitar

low

(s)

VI⁰ 5.4 III⁰ VI⁰ III⁶ VI⁴ VI⁰ 4.7

high

guitar

low

(s)

III⁰ IV⁰ III⁶ VI⁰ 3.9 III⁰ VI⁰ IV⁰ III⁶ VI⁴

high

guitar

low

(s)

VI⁰ 3.1 IV⁰ III⁶ VI⁰ 2.3 III⁰ VI⁰ IV⁰

DRAFT - 2020.07.13

high (40) (8)

guitar (8)

low (8)

III⁶ VI⁴ VI⁰ III⁰ VI⁰ III⁶

1.6

high (44) (8)

guitar (8)

low (8)

III⁰ IV⁰ III⁶ #VI⁴

0.8

8.3

high (47) (8)

guitar (8)

low (8)

IV⁰ II⁰ V⁰ VI⁰ II⁶ #V⁴ VI⁰ II⁴ V⁰ #VI⁴ II⁰ V⁴ VI⁴ V⁰ #VI⁴ IV⁴ V⁴ IV⁴ II⁰

0.0

high (51) (8)

guitar (8)

low (8)

III⁰ V⁰ IV⁴ II⁰ #III⁴ VI⁴ VI⁰ II⁴ VI⁰ VI⁰ II⁰ #VI⁴ II⁰ #VI⁴ VI⁴

high (55) (8)

guitar (8)

low (8)

VI⁴ II⁰ III⁴ VI⁴ IV⁴ I⁰ III⁰ II⁴ IV⁴ I⁰ V⁰ #VI⁴ IV⁴ I⁰ V⁰ VI⁰

59 (8)

high

guitar

low

Chords: I^6 , $\#V^4$, V^3 , VI^0 , IV^0 , I^0 , V^3 , VI^0 , IV^0 , I^5 , V^2 , VI^0 , IV^0 , I^4 , V^2 , $\#V^1$, VI^0 , IV^0 , I^0

63 (8)

high

guitar

low

Chords: V^0 , $\#VI^4$, VI^3 , IV^3 , I^0 , V^0 , VI^0 , IV^3 , I^3 , $\#III^0$, I^1 , II^4 , II^2 , IV^3 , I^1 , IV^0 , I^1 , IV^3 , $\#IV^1$, I^0

67 (8)

high

guitar

low

Chords: $\#IV^1$, I^0 , IV^1 , I^0 , IV^0 , I^0 , IV^0 , I^0 , IV^0 , I^0 , IV^0 , I^0 , IV^0 , II^0 , VI^0 , IV^0 , II^0

3.3

71 (8)

high

guitar

low

Chords: VI^2 , IV^0 , I^0 , IV^0 , I^0 , IV^0 , I^0 , IV^0 , I^0 , IV^0 , I^0 , IV^0 , I^0 , IV^0 , I^0 , IV^0 , I^0 , IV^0

1.2

75 (8)

high

guitar

low

Chords: II^0 , $\#VI^1$, I^0 , IV^0 , II^0 , $\#VI^1$, IV^0 , I^0 , V^0 , III^0 , IV^0 , I^0 , V^0 , III^0 , II^0 , IV^0 , I^0 , V^0 , II^1 , III^4 , III^2

high

guitar

low

Chords: V°, I°, IV°, V°

9.1

high

guitar

low

0.0

5

high

guitar

low

9

high

guitar

low

13

high

guitar

low

17

high

guitar

low

Chord symbols and musical notation are present throughout the system.

21 (8) 9.2

high

guitar

low

(8)

III⁰ V⁵ I⁵ IV⁰ II⁰ III⁵ V⁵ I⁰ IV⁶ V⁰ II⁴ IV⁰ II⁰ III⁵ I⁰ IV⁶ V⁰

7.0

25 (8)

high

guitar

low

(8)

II⁰ III⁵ V⁵ I⁰ IV⁶ V⁰ II⁴ IV⁰ II⁰

6.4 5.8

29 (8)

high

guitar

low

(8)

V⁵ I⁰ V⁰ II⁴ IV⁰ II⁰ V⁵ V⁰ II⁴

5.3

33 (8)

high

guitar

low

(8)

IV⁰ II⁰ III⁵ I⁰ IV⁶ II⁴ IV⁰ II⁰

4.7 4.1

37 (8)

high

guitar

low

(8)

III⁵ V⁵ I⁰ IV⁶ V⁰ IV⁰ II⁰ III⁵

3.5

41 (8)

high

guitar

low

V⁵ I⁰ IV⁶ V⁰ II⁴ IV⁰ II⁰ III⁵ V⁵

2.9

45 (8)

high

guitar

low

I⁰ IV⁶ V⁰ II⁴ IV⁰ III⁵ I⁰

2.3

49 (8)

high

guitar

low

IV⁶ V⁰ II⁴ IV⁰ II⁰ III⁵ V⁵ I⁰

1.8

53 (8)

high

guitar

low

IV⁶ V⁰ IV⁰ II⁰ III⁵ I⁰ II⁴ IV⁰ II⁰

1.2 0.6

57 (8)

high

guitar

low

III⁵ V⁵ I⁰ IV⁶ V⁰ II⁴

9.3

high

guitar

low

61

65

69

73

77

0.0

4.2

1.3

VI⁰ IV⁶ IV⁴ V⁰ II² VI³ IV⁴ V⁰ II² VI⁰ III³ IV² V⁰ III⁰ IV² V⁰ III² IV⁰

V⁰ VI³ VI² V⁰ III² VI⁰ V⁰ III⁰ VI² V³ III⁰ VI⁰ V⁰ III⁰ VI⁰ III² IV⁰ VI⁰ III⁰ V² VI⁰ V⁰

VI² III⁰ V² VI⁰ V² VI² III⁰ V⁰ VI⁰ III² V⁰ VI² V⁰ I⁴ V⁰ III⁰ V² I⁰

II² II¹ V¹ I⁴ I³ II⁰ V⁰ I⁰ II⁰ V⁰ I⁰ II⁰ V⁰ I³ I² II⁰ IV⁰ VI⁰ IV⁰

I⁰ III⁰ IV⁰ I⁰ III⁰ IV⁰ I⁰ III⁰ IV⁰ I⁰ III² I⁰ III⁰ I⁰ III¹

DRAFT - 2020.07.13

21 (8) 10.2

high

guitar

low

5.0

Chords: III⁰, V⁷, VI⁰, V⁷, I⁰, II⁰, VI⁰, V⁷, I⁰, II⁰, V⁷, II⁰, III⁹, #VI⁶, II⁰, VI⁰, V⁷

25 (8) 4.6

high

guitar

low

Chords: II⁰, III⁹, #VI⁶, II⁰, VI⁰, I⁰, II⁰, V⁷, II⁰, III⁹, #VI⁶

29 (8) 4.2 3.8

high

guitar

low

Chords: II⁰, VI⁰, V⁷, II⁰, II⁰, III⁹, #VI⁶, II⁰, VI⁰, V⁷

33 (8) 3.5

high

guitar

low

Chords: II⁰, V⁷, II⁰, III⁹, VI⁶, VI⁰, V⁷, II⁰, #VI⁶

37 (8) 3.1 2.7

high

guitar

low

Chords: II⁰, VI⁰, V⁷, I⁰, II⁰, V⁷, III⁹, II⁰, VI⁰, V⁷

41 (8)

high

guitar

low

2.3

Chord symbols: I^0 , II^0 , V^7 , II^0 , III^9 , II^0 , VI^0 , V^7 , I^0

45 (8)

high

guitar

low

1.9

Chord symbols: II^0 , V^7 , II^0 , $\sharp VI^6$, II^0 , VI^0 , V^7 , I^0 , II^0

49 (8)

high

guitar

low

1.5

Chord symbols: V^7 , III^9 , $\sharp VI^6$, II^0 , V^7 , I^0 , II^0 , V^7

53 (8)

high

guitar

low

1.2

Chord symbols: II^0 , $\sharp VI^6$, II^0 , I^0 , II^0 , II^0 , III^9 , $\sharp VI^6$

57 (8)

high

guitar

low

0.8

0.4

Chord symbols: II^0 , VI^0 , I^0 , II^0 , II^0 , III^9 , $\sharp VI^6$, II^0 , VI^0

high (61) (8)

guitar (8)

low (8)

V⁷ I⁰ II⁰ V⁷ II⁰ III⁹ VI⁶

high (65) (8)

guitar (8)

low (8)

0.0

I⁰ III⁸ I⁰ III⁰ I⁶ III⁰ I⁰ III⁸ I⁰ III⁷ I⁶ III⁷ III⁵ I⁵ III⁰ I⁰ III⁵ I⁵ III⁰ I⁰ III⁵ I⁶

high (69) (8)

guitar (8)

low (8)

II⁸ II⁷ III⁵ I⁴ II⁷ III⁴ I⁴ II⁶ III⁴ I⁰ II⁰ III⁴ I⁴ II⁴ III⁰ I⁴ II⁴ III⁰ I⁴

VI⁰

high (73) (8)

guitar (8)

low (8)

II⁴ III⁰ I⁴ II⁰ III⁴ I⁰ II⁰ III⁰ I⁰ II⁰ III⁰ I⁰ II⁴ III⁰ I⁴ II⁰ III⁰ I⁰ V⁰ III⁴

high (77) (8)

guitar (8)

low (8)

IV⁰ VI⁰ II⁰ V⁷ III⁴ IV⁰ II⁰ V⁵ III⁰ IV⁶ VI⁰ II⁰ V³ III⁰ IV⁰ VI⁶ VI⁴ II⁰ V³

81 (8)

high

guitar

low

4.6

85 (8)

high

guitar

low

1.6

89 (8)

high

guitar

low

93 (8)

high

guitar

low

10.4

0.0

97 (8)

high

guitar

low

101 (8)

high

guitar

(8)

high

guitar

low

(8)

VI⁷ VI⁵ III⁰ V⁰ II⁵ IV⁵ VI⁵ III⁰ V⁷ II⁰ IV⁰ VI⁵

high

guitar

low

(8)

11.2

3

2

5.0

III⁰ IV⁰ VI⁵ IV⁰ I⁷ II⁵ II⁴ VI⁵ III⁰ VI⁵ IV⁰ I⁷ II⁵

high

guitar

low

(8)

4.6

4.2

II⁴ IV⁰ VI⁵ IV⁰ II⁵ II⁴ VI⁵ III⁰ IV⁰ VI⁵ IV⁰ I⁷

high

guitar

low

(8)

3.8

II⁵ II⁴ VI⁵ III⁰ IV⁰ VI⁵ IV⁰ I⁷ II⁵ II⁴

high

guitar

low

(8)

3.3

2.9

2.5

VI⁵ III⁰ IV⁰ VI⁵ II⁵ VI⁵ III⁰ IV⁰ VI⁵ I⁷ II⁴ VI⁵ III⁰ IV⁰

high

guitar

low

(8)

39

VI⁵ IV⁰ I⁷ II⁵ II⁴ VI⁵ III⁰ IV⁰ VI⁵ IV⁰

2.1

high

guitar

low

(8)

43

II⁵ II⁴ VI⁵ III⁰ IV⁰ VI⁵ IV⁰ I⁷ II⁵ VI⁵

1.7 1.3

high

guitar

low

(8)

47

III⁰ IV⁰ VI⁵ I⁷ II⁵ II⁴ VI⁵ III⁰ IV⁰ VI⁵

0.8

high

guitar

low

(8)

51

IV⁰ I⁷ II⁵ II⁴ VI⁵ III⁰ IV⁰

0.4

high

guitar

low

(8)

54

VI⁵ IV⁰ I⁷ II⁵ III⁰ IV⁵ IV⁴ III⁰ IV⁰ III⁰ IV⁴ III⁰ IV⁰ VI⁰ V⁷

11.3 2/2 0.0

57 (8)

high

guitar

low

Chords: V^5 , II^0 , I^5 , IV^0 , III^0 , IV^4 , III^0 , IV^4 , III^6 , IV^0 , III^0 , IV^0 , III^0 , IV^0 , III^0 , IV^0 , III^0 , IV^0 , III^0 , IV^3 , III^0 , IV^3

VI⁰, VI⁴, VI³, VI⁰, VI⁰, VI⁰

61 (8)

high

guitar

low

Chords: VI^0 , II^3 , I^6 , V^3 , III^6 , II^0 , I^0 , V^3 , III^0 , II^0 , I^0 , V^0 , III^0 , II^0 , I^6 , I^4 , II^3 , IV^3

65 (8)

high

guitar

low

Chords: III^0 , I^3 , II^3 , IV^3 , III^0 , I^0 , II^0 , IV^0 , III^0 , I^3 , II^0 , IV^0 , III^0 , I^0 , II^0 , IV^3 , III^0 , I^3 , II^0 , IV^3 , III^0

69 (8)

high

guitar

low

Chords: II^3 , V^0 , II^3 , V^0 , VI^0 , V^0 , VI^0 , V^0 , I^3 , VI^3 , V^3 , II^3 , I^3 , VI^3 , V^0 , II^0 , I^3 , VI^0 , V^3

73 (8)

high

guitar

low

Chords: II^3 , VI^3 , V^3 , I^0 , IV^0 , VI^3 , II^0 , VI^3 , II^0 , V^3 , I^0 , III^0 , VI^3 , IV^3 , V^3 , I^0

77 (8)

high

guitar

low

III⁶ III⁵ IV⁰ V⁰ I⁰ III⁴ VI⁹ VI¹ IV⁰ I⁰ IV² VI¹ V⁰ IV⁰ VI¹ V¹ IV⁰ VI⁰ V⁰ IV⁰ II¹

81 (8)

high

guitar

low

I² VI⁰ IV⁰ V⁰ I⁰ II¹ IV⁰ VI⁰ V⁰ I² II⁰ IV⁰ VI⁰ IV⁰ I⁰ II⁰ V⁰ IV⁰ I⁰ II⁰ V⁰

4.8

85 (8)

high

guitar

low

IV⁰ I¹ II⁰ V⁰ VI⁰ V⁰ III³ II⁰ IV⁰ VI⁰ V⁰ III⁰ II⁰ IV⁰ V⁰ IV⁰ III⁰ I⁰ II⁰ VI⁰ I⁰

1.5

89 (8)

high

guitar

low

III² VI⁰ IV⁰ II⁰ V⁰ III² V⁰ II⁰ VI⁰ III⁰ IV⁰ V⁰ II⁰ VI⁰ IV⁰ II⁰ V⁰ VI⁰ IV⁰ I⁰ II⁰

11.4

93 (8)

high

guitar

low

III⁰ IV⁰ I⁰ II⁰ III⁰ IV⁰ I⁰ II⁰ III⁰ IV⁰ I⁰ II⁰ III⁰

0.0

97 (8)

high

guitar

(8)

IV°

I°

II°

III°

low

(8)

DRAFT - 2020.07.13

high

guitar

low

(21) (8)

IV⁰ V⁰ III⁶ IV⁰ V⁰ III⁶ IV⁰ V⁰ III⁶ IV⁰ V⁰ III⁶ IV⁰ V⁰ III⁶ IV⁰ V⁰ VI⁹

high

guitar

low

(25) (8)

VI⁷ III⁶ I⁶ V⁰ VI⁶ III⁰ I⁰ V⁰ VI⁰ III⁶ I⁶ V⁶ VI⁰ III⁶ I⁰ V⁶ VI⁶ I⁰

high

guitar

low

(29) (8)

VI⁶ III⁶ I⁰ V⁰ VI⁰ III⁶ I⁶ V⁰ VI⁰ III⁶ I⁰ V⁶ II⁰ VI⁶ IV⁰ III⁰ II⁰ VI⁶ IV⁰

high

guitar

low

(33) (8)

III⁰ II⁰ VI⁶ IV⁰ III⁰ II⁰ VI⁰ III⁶ II⁰ IV⁰ III⁶ V⁶ III⁰ II⁰ V⁰ III⁶ II⁷

high

guitar

low

(36) (8)

12.2

5.0 4.4

V⁰ III⁵ II⁵ V⁰ III⁵ III³ II⁵ III⁵ III³ III⁵ V⁰

39 (8)

high

guitar

low

3.9

3.3

III⁵ III⁵ III⁵ II⁵ V⁰ III⁵ III⁵

43 (8)

high

guitar

low

2.8

2.2

III⁵ II⁵ V⁰ III⁵ III³ III⁵ II⁵

47 (8)

high

guitar

low

1.7

V⁰ III⁵ III³ III⁵ II⁵ V⁰ III⁵ III³

51 (8)

high

guitar

low

1.1

0.6

III⁵ II⁵ V⁰ III⁵ II⁵ III⁵

12.3

55 (8)

high

guitar

low

0.0

II⁴ IV⁰ V⁰ I⁴ III³ VI⁰ IV⁰ V⁰ I⁰ III⁰ VI⁶ IV⁵ V⁰ I⁰ III⁰ VI⁰ IV⁴ V⁵

59 (8)

high

guitar

low

Chords: I⁰, III⁰, IV⁰, V⁰, I⁴, I³, III³, VI⁰, IV⁰, V⁰, I⁰, III³, #VI⁶, IV⁰, V⁰, I⁰, III⁰, VI⁰, IV⁰

63 (8)

high

guitar

low

Chords: V⁴, I⁰, III³, VI⁰, V⁰, I³, VI⁰, V⁰, II⁰, VI⁵, V⁰, II⁰, VI³, V⁴, II⁰, VI⁰, V⁰, II²

67 (8)

high

guitar

low

Chords: I⁰, V³, I⁰, #V¹, I⁰, V⁰, I⁰, V⁰, I⁰, V⁰, I⁰, VI³, V⁰, II⁰, IV⁰, I⁰, VI², V⁰, II⁰, IV², I⁰

3.2

71 (8)

high

guitar

low

Chords: VI⁰, V⁰, II⁰, IV², IV⁰, I⁰, VI⁰, V⁰, II⁰, IV⁰, I⁰, III¹, IV⁰, VI⁰, I⁰, V⁰, VI⁰, V⁰, IV⁰, II⁰, I⁰

2.1

75 (8)

high

guitar

low

Chords: III¹, V⁰, VI⁰, V⁰, VI⁰, II⁰, IV⁰, III⁰, V⁰, III⁰, V⁰, III⁰, V⁰, III⁰

12.4

0.0

high

guitar

low

(78) (8)

V° III° V° III° V° III° V° III°

high

guitar

low

(82) (8)

II° I° VI°

13.1

high

guitar

low

0.0

5

high

guitar

low

9

high

guitar

low

13

high

guitar

low

13.2

7.0

17

high

guitar

low

The musical score is divided into five systems. Each system contains three staves: 'high' (treble clef), 'guitar' (treble clef with chord diagrams), and 'low' (bass clef). The guitar part is heavily annotated with chord diagrams such as III⁰, IV¹¹, IV¹⁰, I⁰, III⁰, IV⁰, V¹⁰, V⁸, I⁹, I⁸, III⁰, IV¹⁰, V⁸, I⁸, III⁰, IV⁰, II⁰, IV⁹, IV⁸, II⁰, IV⁰, II⁸, I⁰, IV⁰, VI¹⁰, II⁰, I⁰, IV⁰, VI⁰, II⁸, III⁹, IV⁰, II⁰, VI⁰, V⁰, II⁸, II⁷, VI⁰, V⁷, II⁰, VI⁸, V⁷, V⁶, III⁰, IV⁸, I⁸, III⁹, IV⁰, II⁰, VI⁰, V⁰, II⁷, V⁷, VI⁰, and VI⁸. Measure numbers 5, 9, 13, 13.2, and 17 are indicated at the start of their respective systems. A large 'DRAFT' watermark is visible across the top left of the page.

high (21) (8)

guitar (8) V^6 III^0 IV^8 II^8 III^9 IV^0 II^0 V^0 II^8 II^7 V^7

low (8) 6.3

high (25) (8)

guitar (8) II^0 VI^8 V^7 V^6 III^0 I^8 II^8 III^9 VI^0

low (8) 5.6

high (29) (8)

guitar (8) II^8 II^7 VI^0 V^7 VI^8 V^7 V^6 III^0 IV^8 I^8 II^8 IV^0

low (8) 4.9

high (33) (8)

guitar (8) II^0 VI^0 V^0 II^8 VI^0 V^7 II^0 VI^8 V^7 V^6

low (8)

high (37) (8)

guitar (8) III^0 IV^8 I^8 II^8 III^9 II^0 VI^0 V^0 II^7 VI^0

low (8) 4.2

41 (8)

high

guitar

low

3.5

VI⁸ V⁶ III⁰ I⁸ IV⁰ II⁰ VI⁰ V⁰ II⁸ VI⁰ V⁷

45 (8)

high

guitar

low

2.8

II⁰ VI⁸ V⁷ V⁶ III⁰ IV⁸ I⁸ II⁸ III⁹ IV⁰

49 (8)

high

guitar

low

VI⁰ V⁰ II⁸ II⁷ V⁷ II⁰ VI⁸ V⁷

53 (8)

high

guitar

low

2.1

V⁶ I⁸ II⁸ II⁰ VI⁰ II⁸ VI⁰ V⁷ II⁰ VI⁸

57 (8)

high

guitar

low

1.4

V⁷ V⁶ IV⁸ I⁸ II⁸ III⁹ IV⁰ VI⁰ V⁰

high

guitar

low

(61) (8)

0.7

Chords: Π^6 , Π^7 , VI^0 , V^7 , Π^0 , $\#VI^8$, V^7 , $\#V^6$, III^0

high

guitar

low

(65) (8)

Chords: IV^8 , I^6 , Π^6 , III^9 , Π^0 , VI^0 , Π^6 , Π^7 , Π^0 , $\#VI^8$

high

guitar

low

(69) (8)

13.3

0.0

Chords: V^7 , V^6 , Π^6 , VI^0 , V^6 , Π^0 , VI^7 , III^0 , IV^8 , IV^6 , III^9 , Π^0 , V^0 , IV^0 , III^9 , Π^0 , V^6 , IV^0

high

guitar

low

(73) (8)

Chords: Π^0 , IV^6 , Π^0 , IV^6 , I^8 , $\#VI^6$, IV^0 , I^0 , VI^0 , IV^0 , Π^6 , III^8 , VI^0 , V^0 , III^8 , $\#III^6$, VI^5 , VI^5 , V^0 , IV^6

high

guitar

low

(77) (8)

Chords: VI^5 , III^0 , Π^0 , I^6 , Π^5 , III^0 , IV^6 , IV^5 , Π^4 , III^5 , IV^5 , IV^3 , III^4 , IV^0 , III^3 , IV^3 , V^6 , V^5 , I^6 , I^5 , III^0 , VI^0

81 (8)

high

guitar

low

(8)

V⁰ I³ III⁰ VI⁰ V⁰ I⁰ III⁰ VI⁰ V⁰ I² III⁰ VI⁰ V⁰ I⁰ III⁰ VI⁵ VI³ V⁵ V³

85 (8)

high

guitar

low

(8)

I² I¹ III³ III¹ VI² V³ #V¹ I¹ III¹ #VI¹ V⁰ I⁰ III¹ VI⁰ V⁰ I⁰ V⁰ II⁰

89 (8)

high

guitar

low

(8)

I⁰ III¹ V⁰ III¹ II⁴ I⁰ VI⁰ I⁰ II⁰ V⁰ VI⁰ I⁰ II⁴ V⁰ VI⁰ I⁰ II³ V⁰ VI⁰ I⁰ II³ V⁰ VI⁰ I⁰

93 (8)

high

guitar

low

(8)

II⁰ V⁰ III¹ III⁰ II⁰ V⁰ III⁰ II³ II¹ V⁰ III⁰ II⁰ V⁰ IV⁰ II⁰ III⁰ IV⁰ V⁰ VI⁰

4.9

97 (8)

high

guitar

low

(8)

II⁰ III⁰ IV⁰ V⁰ VI⁰ II⁰ III⁰ IV⁰ V⁰ VI⁰ V⁰ III⁰ I⁰ II⁰ VI⁰ V⁰ IV³ III⁰ II⁰ VI⁰

101 (8)

high

guitar

low

2.4

Chord progression for measures 101-104:

- Measure 101: V⁰
- Measure 102: IV⁰
- Measure 103: III⁰
- Measure 104: II⁰

105 (8)

13.4

high

guitar

low

0.0

Chord progression for measures 105-108:

- Measure 105: IV⁰
- Measure 106: III⁰
- Measure 107: II⁰
- Measure 108: I⁰

109 (8)

high

guitar

low

Chord progression for measures 109-112:

- Measure 109: III⁰
- Measure 110: II⁰
- Measure 111: V⁰
- Measure 112: VI⁰

high

guitar

low

(8)

(8)

(8)

III⁰ I⁹ II⁹ V⁰ IV⁰ III⁹ I⁹ II⁰ V⁰ IV⁹ III⁹ I⁰ II⁹ V⁰ VI¹⁰ V⁰ III⁰

high

guitar

low

(21) (8)

I⁰ V⁵ II⁰ V⁰ II⁶ II⁵ V⁰ II⁴ V⁰ VI⁰ IV⁵ I⁰ V⁵ V⁴ VI⁰ IV⁰ I⁰

high

guitar

low

(25) (8)

14.2

V⁴ II⁰ IV⁰ I⁰ III⁵ IV⁰ II⁴ VI⁵ II³ V⁴ VI⁰ IV⁰ I⁰ V⁴ II⁰

7.0

high

guitar

low

(29) (8)

IV⁰ I⁰ III⁵ IV⁰ II⁴ VI⁵ II³ VI⁰ I⁰ V⁴ II⁰

5.6

high

guitar

low

(33) (8)

IV⁰ I⁰ IV⁰ VI⁵ II³ IV⁰ I⁰ V⁴ II⁰

4.2

high

guitar

low

(37) (8)

IV⁰ III⁵ IV⁰ II⁴ VI⁵ II³ V⁴ IV⁰ I⁰ V⁴ II⁰

2.8

high

guitar

low

(41) (8)

IV⁰ I⁰ III⁵ II³ V⁴ VI⁰ IV⁰ I⁰ V⁴

1.4

high

guitar

low

(45) (8)

II⁰ IV⁰ I⁰ III⁵ IV⁰ VI⁵ II³

14.3

V⁰ III⁴ I⁰ III⁰ VI⁰

0.0

high

guitar

low

(49) (8)

VI⁵ I⁰ III⁰ VI⁰ I⁶ III⁴ III³ I⁰ III³ VI⁵ VI³ I⁰ III⁰ VI³ I⁴ III⁰ VI⁰ I⁰ III² VI³

high

guitar

low

(53) (8)

VI² I⁰ V² VI⁰ I² IV⁰ VI⁰ II⁰ III² IV⁰ VI⁰ II⁰ III² IV⁰ VI⁰ II⁰

high

guitar

low

(57) (8)

III⁰ II⁰ V² I² VI² IV⁰ II⁰ V⁰ I² VI⁰ IV⁰ II³ II² IV⁰ I⁰

high

guitar

low

(61) (8)

II² I⁰ V² III⁰ VI² II² I⁰ V⁰ III⁰ VI² II² I² V⁰ III⁰ VI² II⁰ I² V² III⁰ VI⁰ II⁰ I⁰ V⁰

high

guitar

low

(65) (8)

III⁰ VI⁰ II⁰ I² V² III⁰ VI² II⁰ I² V⁰ III⁰ VI⁰ IV⁰ V² VI⁰ II⁰ I⁰ IV³ V⁰ VI⁰

4.4

high

guitar

low

(69) (8)

II⁰ I⁰ IV³ IV¹ V⁰ VI⁰ II⁰ I⁰ IV¹ IV⁰ V⁰ VI² II⁰ I⁰ IV⁰ V⁰ VI² II⁰

2.7

high

guitar

low

(73) (8)

I¹ III⁰ VI⁰ IV⁰ II⁰ III⁰ VI⁰ IV⁰ II⁰ III⁰ VI⁰ IV⁰ II⁰

14.4

0.0

high

guitar

low

(77) (8)

V⁰ III⁰ I⁰ II⁰ V⁰ III⁰ IV⁰ III⁰ V⁰

DRAFT

high

guitar

low

(81) (8)

(8)

IV°

VI°

III°

The image shows a musical score for guitar, divided into high and low registers. The high register is on a treble clef staff, and the low register is on a bass clef staff. The guitar part is written on a single staff between the two registers. The score includes a key signature of one sharp (F#) and a time signature of 4/4. The high register part starts with a whole note chord (F#4, A4, C5) and a half note chord (F#4, A4, C5). The guitar part starts with a whole note chord (F#4, A4, C5) and a half note chord (F#4, A4, C5). The low register part starts with a whole note chord (F#4, A4, C5) and a half note chord (F#4, A4, C5). The score ends with a double bar line.

The image displays a musical score for the song "The Sound of Silence" by Simon & Garfunkel. It features three staves: a high staff (treble clef), a guitar staff (treble clef), and a low staff (bass clef). The guitar staff includes chord diagrams and chord names (VI⁷, V⁶, V⁷, III⁷, I⁰, VI⁰, V⁰, III⁷, V⁰, IV⁰, VI⁷, II⁰, III⁷, V⁰, IV⁰, VI⁷, II⁸, III⁷, V⁰, IV⁸, VI⁷) and a key signature of one sharp (F#). The low staff is mostly empty, with a few notes in the first measure. The high staff has a few notes in the first measure. The guitar staff has a key signature of one sharp (F#) and a tempo/mood marking of "Moderato".

21 (8)

high

guitar

low

Chords: Π^0 , III^0 , V^0 , IV^7 , VI^0 , Π^0 , III^7 , V^0 , IV^0 , VI^0 , Π^0 , III^0 , V^0 , IV^0 , VI^7 , II^7 , III^7 , V^7 , IV^0 , Π^0 , III^0

25 (8)

high

guitar

low

Chords: V^0 , IV^0 , VI^7 , Π^7 , III^7 , V^7 , IV^7 , VI^7 , Π^7 , IV^0 , I^8 , VI^0 , V^5 , IV^6 , III^0 , IV^0 , III^7 , III^0 , VI^0 , VI^0

29 (8)

high

guitar

low

Chords: IV^5 , V^5 , I^0 , III^7 , VI^5 , IV^0 , V^0 , I^0 , III^0 , VI^0 , IV^0 , V^5 , I^6 , III^7 , VI^4 , IV^4 , V^0 , I^0 , III^0 , I^0 , V^0

33 (8)

high

guitar

low

Chords: IV^0 , III^6 , I^0 , V^0 , IV^4 , V^0 , IV^4 , II^6 , III^6 , VI^3

15.2

Chords: VI^4 , IV^4 , V^0

7.0

36 (8)

high

guitar

low

Chords: I^0 , I^0 , V^0 , IV^0 , III^6 , V^0 , IV^4 , IV^4 , III^6 , VI^3

high

guitar

low

6.0

(8)

(40)

(8)

V⁰ III⁰ V⁰ IV⁰ III⁶ I⁰ V⁰ IV⁴ V⁰ II⁶ III⁶

high

guitar

low

5.0

(8)

(44)

(8)

VI³ VI⁴ V⁰ I⁰ III⁰ V⁰ IV⁰ III⁶ I⁰

high

guitar

low

4.0

(8)

(48)

(8)

V⁰ IV⁴ V⁰ II⁶ III⁶ VI³ VI⁴ IV⁴ V⁰

high

guitar

low

(8)

(52)

(8)

I⁰ III⁰ I⁰ V⁰ IV⁰ I⁰ IV⁴ V⁰ II⁶ III⁶

high

guitar

low

3.0

(8)

(56)

(8)

VI⁴ V⁰ I⁰ III⁰ V⁰ IV⁰ III⁶ I⁰ V⁰

high

guitar

low

(60) (8)

V⁰ IV⁴ II⁶ III⁶ VI³ VI⁴ IV⁴ V⁰ I⁰

2.0

high

guitar

low

(64) (8)

III⁰ V⁰ IV⁰ III⁶ I⁰ IV⁴ V⁰

high

guitar

low

(68) (8)

IV⁴ II⁶ III⁶ VI³ IV⁴ I⁰ III⁰ IV⁰ I⁰

1.0

high

guitar

low

(72) (8)

V⁰ IV⁴ V⁰ IV⁴ III⁶ VI³

high

guitar

low

15.3

(76) (8)

V⁴ IV⁰ II⁰ III⁰ III⁶ III⁵ I⁴ V⁰ I⁰ V³ I⁰ IV⁰ I⁰ IV⁴ IV³ I⁰ IV³ I⁴ IV³ I⁰ IV³ I⁰ IV³ I⁰

0.0

high

guitar

low

(80) (8)

IV⁰ I⁰ III⁵ III⁴ VI³ IV³ I⁰ V⁰ III⁰ V⁰ I⁴ I³ III⁰ V⁰ I⁵ III⁰ V⁰ I⁰ III⁰ V⁰ I³ III⁰ V³ I⁰

high

guitar

low

(84) (8)

III⁰ V⁰ VI³ II⁵ V⁰ IV⁰ V⁰ II⁰ III⁰ I³ II⁰ I⁰ VI⁰ V⁰ IV⁰ V⁰ III⁰ II⁴ IV⁰

3.9

high

guitar

low

(88) (8)

I³ V³ III⁰ II⁰ IV⁰ I⁰ V² III⁰ II⁰ IV² I⁰ # V¹ III⁴ III² II⁰ IV⁰ I² # V¹ III⁰ II²

1.3

high

guitar

low

(92) (8)

IV¹ I⁰ V⁰ III⁰ II¹ IV⁰ I⁰ V⁰ III⁰ II⁰ IV⁰ I⁰ VI³ VI¹ II⁰ III⁰ V⁰ I⁰

high

guitar

low

(96) (8)

15.4

II⁵ III⁵ V⁵ I⁵ V⁵ I⁵ II⁵ IV⁵ V⁵ I⁵ II⁵ IV⁵

0.0

DRAFT

high

guitar

low

(100) (8)

V° IV° I° III°

high

guitar

low

(103) (8)

VI°

DRAFT - 2020.07.13

high

guitar

low

(21) (8)

II⁰ IV⁵ III⁵ I⁰ V⁰ III⁵ I⁵ V⁰ III⁰ V⁰ VI⁰ III⁰ IV⁰ V⁸ V⁷ VI⁰ III⁰ IV⁴

high

guitar

low

16.2

(25) (8)

II⁰ IV⁵ III⁵ I⁰ III⁵ I⁵ V⁰ III⁰ V⁰ VI⁰ IV⁰

5.0

high

guitar

low

(29) (8)

V⁸ V⁷ III⁰ IV⁴ II⁰ IV⁵ I⁰ VI⁰

4.6

high

guitar

low

(33) (8)

I⁵ III⁰ V⁰ VI⁰ IV⁰ V⁸ V⁷ III⁰

high

guitar

low

(37) (8)

IV⁴ II⁰ IV⁵ III⁵ V⁰ III⁵ I⁵ V⁰

4.2

41 (8)

high

guitar

low

3.8

45 (8)

high

guitar

low

49 (8)

high

guitar

low

3.3

53 (8)

high

guitar

low

57 (8)

high

guitar

low

2.9

61 (8)

high

guitar

low

V⁰ I⁵ V⁰ III⁰ V⁰ VI⁰ III⁰ IV⁰ IV⁴

65 (8)

high

guitar

low

II⁰ IV⁵ III⁵ V⁰ III⁵

2.5

69 (8)

high

guitar

low

I⁵ V⁰ III⁰ V⁰ VI⁰ III⁰ IV⁰ V⁸ V⁷ VI⁰ II⁰ VI⁰

2.1

73 (8)

high

guitar

low

IV⁵ V⁰ III⁵ I⁵ V⁰ V⁰ VI⁰ IV⁰ V⁸

77 (8)

high

guitar

low

V⁷ VI⁰ III⁰ III⁵ I⁰ V⁰ III⁵

1.7 1.7

81 (8)

high

guitar

low

Chords: I^5 , III^0 , V^0 , VI^0 , IV^0 , V^8 , V^7 , VI^0 , IV^4

85 (8)

high

guitar

low

Chords: II^0 , VI^0 , III^5 , I^0 , V^0 , III^5 , I^5 , III^0

1.3

89 (8)

high

guitar

low

Chords: V^0 , VI^0 , IV^0 , V^8 , V^7 , VI^0 , III^0 , II^0 , VI^0

0.8

93 (8)

high

guitar

low

Chords: III^5 , I^0 , V^0 , III^5 , I^5 , V^0 , III^0 , V^0 , VI^0

97 (8)

high

guitar

low

Chords: III^0 , IV^0 , V^8 , V^7 , III^0 , IV^4

101 (8)

high

guitar

low

0.4

VI⁰ IV⁵ V⁰ III⁵ I⁵ V⁰ III⁰

105 (8)

high

guitar

low

V⁰ VI⁰ III⁰ IV⁰ V⁸ V⁷ VI⁰ III⁰ IV⁴

16.3

2/2

109 (8)

high

guitar

low

0.0

I⁰ V⁷ III⁰ II⁵ II⁴ IV⁰ V⁷ III⁴ VI⁰

112 (8)

high

guitar

low

II⁴ IV⁰ V⁰ III⁴ VI⁴ IV⁴ V⁰ IV⁴ I⁰ IV⁰ I⁰ V⁰ IV⁴ III⁰ V⁰ IV⁰ III⁴ I⁰ II⁴ III⁴ V⁰

116 (8)

high

guitar

low

III⁴ I⁰ II⁴ V⁰ III⁰ I⁰ II⁰ IV⁰ VI⁴ III⁰ V⁷ V⁶ IV⁰ III⁰ IV⁰ III⁰ IV⁴ III⁰ IV⁰ III⁴

high

guitar

low

(120) (8)

VI⁴ V⁰ I⁵ IV⁰ V⁰ III³ IV⁰ V⁰ VI⁹ VI⁰ V⁰ IV⁰ VI⁰ V⁰ IV⁴ IV³ VI³ V⁰ IV³ I⁵ I³

high

guitar

low

(124) (8)

IV³ I⁰ IV⁰ I³ IV³ I³ IV⁰ III⁰ IV⁰ V⁰ II⁰ IV⁰ VI⁰ III⁰ VI⁰ V⁰ IV³ III³ I⁰ II⁴

high

guitar

low

(128) (8)

VI⁰ V⁰ IV⁰ II³ V⁰ IV⁰ II³ V⁰ IV⁰ II³ V⁰ IV⁰ III³ I³ III³ I³ III⁰ I³ III⁰ I⁰

high

guitar

low

(132) (8)

V⁰ II³ IV⁰ III³ I³ V⁰ II⁰ IV⁰ III⁰ I³ V⁰ II³ IV³ III⁰ I³ V⁰ II⁰ IV⁰ III⁰ I⁰

high

guitar

low

(136) (8)

V⁵ II² IV³ III⁰ I⁰ V⁰ II² IV⁰ III⁰ I² V⁴ II⁰ IV² III⁰ I¹ V³ II⁰ IV¹ III⁰ I⁰

3.9

140 (8)

high

guitar

low

2.7

Chord progression for measures 140-143:

- Measure 140: V⁰, II⁰, IV⁰, III¹, I⁰
- Measure 141: V³, V², II⁰, IV⁰, VI⁰
- Measure 142: II⁰, III⁰, I⁰, V⁰, VI⁰
- Measure 143: II⁰, III⁰, I⁰, #V¹, VI⁰

144 (8)

high

guitar

low

3

7

Chord progression for measures 144-147:

- Measure 144: I⁰, V⁰, VI², V⁰
- Measure 145: IV⁰, V⁰, II⁰, IV⁰, VI⁰
- Measure 146: II⁰, I⁰, #VI¹, II⁰, I⁰
- Measure 147: VI¹, I⁰

16.4

147 (8)

high

guitar

low

0.0

Chord progression for measures 147-150:

- Measure 147: VI⁰, IV⁰, II⁰, V⁰, III⁰, I⁰
- Measure 148: IV⁰, II⁰, V⁰, III⁰, I⁰
- Measure 149: V⁰, III⁰, I⁰, IV⁰, II⁰
- Measure 150: VI⁰

151 (8)

high

guitar

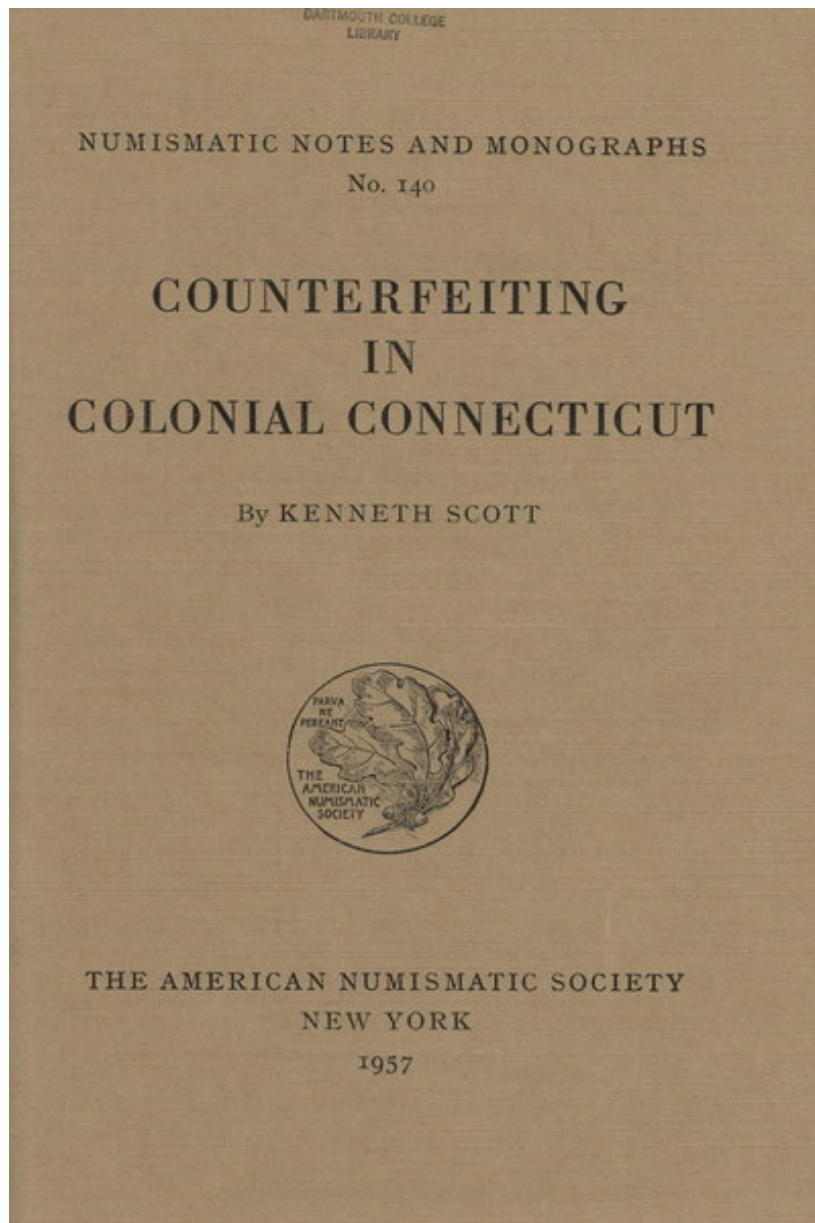
low

Chord progression for measures 151-154:

- Measure 151: I⁰, V⁰, VI⁰
- Measure 152: (Empty)
- Measure 153: (Empty)
- Measure 154: (Empty)

appendix 1 - excerpts from “Counterfeiting in Colonial Connecticut” by Kenneth Scott
reprinted from <http://numismatics.org/digitalibrary/ark:/53695/nnan72127>

Note that there are generally 3 denominations: pounds, shillings, and pence. In the book, pounds are denoted with the prefix “£”. Numbers followed by the suffix “s.” denote shillings. Generally, pence do not occur alone but rather as part of a cumulative sum in the form of pounds/shillings/pence; e.g. “£5/11/9” would be read “five pounds, eleven shillings, and nine pence”. Sometimes 2 numbers instead of 3 are separated by a forward slash. If they are preceded by a “£”, the numbers denote pounds/shillings (e.g. “£3/10s.”). Otherwise the pair denotes shillings/pence as with the common 2/6 which would be pronounced “two shillings and sixpence”.



DRAFT - 2020.07.13

William Barker and Samuel Munn

Early in January, 1712, William Barker and Samuel Munn, who were thought to have come together from Okinoke to Milford, were at Mr. Richard Bryant's house, where Munn paid the reckoning. Both men passed altered bills of credit, and on January 8 a complaint against Barker was made to John Ailing, assistant, at Guilford, who at nine o'clock that evening ordered a hue and cry after Barker, who was said to be a trader from Rhode Island and was thus described: "of red hair, a well made portly man, black wigg, light collour'd loose Coat, dark Colour'd straight Coat, speckled vest dark Colour'd stock Stockings washt leather Breeches who is Charged with ye Crime of Counterfeiting or altering a five Shillings bill of this Colony to five pounds..."

The object of the hue and cry was apprehended at Lyme the next day and was taken before Captain Ely, J.P., of that town, who, after examining the prisoner, ordered the constable of Saybrook to take him to New Haven. On the road Barker broke away but was soon retaken and brought again before Justice Ely. The magistrate now ordered the captive's portmanteau searched, and in Barker's pocket-book were found three counterfeit bills, one of 3s. made into £5, one of 3s. altered to 20s., and one of 2s. raised to 10s.

This paper money was sealed up by the justice, and the criminal was sent off again under guard to New Haven, where he was examined by Warham Mather, J.P. It was discovered that Barker had stopped at a tavern in Killingworth, at Eastchester and at the house of Abraham Chanker, to whom he had passed a counterfeit 10s. bill to pay a reckoning of about 1/8. He likewise had uttered to Tavernkeeper Stiles in Milford a counterfeit 10s. Connecticut bill, no. 3931, which is preserved in the Connecticut State Library.

Justice Mather and John Ailing committed their prisoner to the jail in New Haven on January 11 but three days later, as was reported by Sheriff Joshua Hotchkiss, Barker broke prison and made his escape despite a vigorous pursuit, in the course of which three men set out from Branford in the hope of overtaking the fugitive, Seth Morse and John Hoadly to Guilford and Jacob Carter to Killingworth, all under the supervision of Constable Isaac Foot.

Barker's acquaintance, Samuel Munn of Woodbury, was not as fortunate as his companion. At Milford on January 7, 1712, Samuel Eells, assistant, acting on a complaint lodged by Samuel Stone of that town, issued a warrant to Deputy Sheriff Gideon Buckingham to arrest Munn. Stone charged that on the morning of January 7 at the house of Edward Wilkinson in Milford Munn offered a Connecticut 5s. bill altered to £5 to Wilkinson, who refused it, and then to Samuel Clark, Jr. About nine or ten o'clock John and Samuel Stone arrived and together with Wilkinson pointed out to Munn that the bill was altered. Munn told them that he had received it from Samuel Hawley, Sr., of Stratford and that he would go at once to Stratford to induce Hawley to take back the bill.

Munn was apprehended the same day that the warrant was issued and he was examined before Justice Eells and Jonathan Law, J.P. At first he told the magistrates that he got the counterfeit bill from "old Mr. Samll Hauley," who, he explained, owed him £5 and sent the money by Jonathan Stiles to Francis Stiles, who delivered it to him (Munn). He intended, in case he could not pass the bill in Milford, to destroy or burn it. Finally, however, he confessed that he had bought the bill for 40s. from a stranger from Long Island at Mr. Richard Bryant's house.

Munn was bound over to the next Superior Court to be held at New Haven on the second Tuesday in March but was released on bail provided by Daniel Munn and Ephraim Warner. His sureties brought him into court, where he was indicted for having on January 5 altered a 2s. Connecticut bill to 10s. and passed it to John Camp of Milford; also for having on January 7 altered a 5s. Connecticut bill to £5 (Plate VII) and passed it to Samuel Clark. The witnesses against him were Sergeant John Camp, Edward Elberton, mariner, Edward Wilkinson, Samuel Clark, John Stone and Gamaliel Prime. He pleaded not guilty, was tried, convicted and sentenced to be imprisoned for six months and to pay a fine of £45. The informer against him was granted the reward of £20 established by law.

Barker, doubtless encouraged by his escape, continued his career of crime but on November 15, 1717, made the mistake of passing a counterfeit 20s. bill of Rhode Island to Captain John Raymond, Jr., in Norwalk. Raymond quickly detected the cheat and sent his son after Barker, while he himself hastened to make a complaint to Major Peter Burr, J.P., of Fairfield.

The suspected counterfeiter was soon seized and, when he was searched, two more false 20s. Rhode Island bills were found on him, as well as three 5s. Connecticut bills, a half crown Connecticut bill, three 10s. Boston bills, one 10s. Rhode Island bill and one 5s. and one is. Boston bill. At his examination before Major Burr on November 16 he said that he was from Rhode Island, where he had a father and brothers. He had, he admitted, been in jail in New York and his father had sent £70 there to redeem him. About five years before, he confessed, he had escaped from the jail in New Haven in order to save his life, as he was like to freeze to death. He added that he had not been to Rhode Island for thirteen months and that he came last from the Widow Mead's at Horseneck on Long Island. As for the counterfeit bills, he claimed that he had received two of the 20s. bills from Charles Congrove at the Oyster Pond on Long Island and that he had changed silver with a Hartford man for two 20s. bills. He planned, he said, to obtain money from his father to buy land for a settlement in the "New Country."

Justice Burr was not favorably impressed, especially when a bill, not quite finished, was found in Barker's pocket, so he ordered the prisoner committed to the jail in Fairfield. On the night of November 20, however, Barker broke out but this time was recaptured and returned to prison on November 23 by John Bagly and Lieutenant John Taylor. Now he was confined in irons.

At the Superior Court held in Fairfield on December 11, 1717, Barker, described as late of Newport, Rhode Island, was indicted for having, about November 15 at Norwalk, counterfeited four 20s. Rhode Island bills and for having uttered one of them. He pleaded not guilty, was tried, convicted and at the next sessions of the court on February 5, 1718, was sentenced forthwith to be given thirty lashes on the naked body and again, during the first week in March, to receive another thirty stripes. In addition he was to be imprisoned for six months and to pay costs of £14/0/4. The informer had some difficulty in obtaining his reward, as well as his 20s. and the treble damages due him by law. He therefore memorialized the Assembly in May, 1718, and was granted the reward of £20.

Shubael Rowly, Jr.

At the Superior Court held in New London on March 25, 1712, Shubael Rowly, Jr., of Colchester was supposed to appear. In the latter part of November, 1711, he had altered three Connecticut bills, one of 2s. to 20s., one of 3s. to 30s. and one of 5s. to 50s. The first he passed to Thomas Atwell in New Haven, the second to Sergeant Strickland and the third to Richard Christophers. Christophers at once detected the cheat and bound Rowly over to appear at the next Superior Court. Shubael Rowly, Sr., and Joshua Hempstead provided bail in the amount of £40. The following day, according to Christophers, young Rowly confessed that he had altered the bills and had passed two of them, of which he had taken up one and was desirous of taking up the other.

At the March session of the Superior Court Rowly was called three times but neither he nor his sureties appeared. He was, however, indicted, his bail was declared forfeited, and a warrant was issued for his arrest. Sometime later, probably in September, John Reed, the Queen's Attorney, recovered from Shubael Rowly, Sr., £36 of the forfeited bond and, apparently because of a deal between Reed and the father of young Rowly, the Assembly in October, 1712, was persuaded to pass a resolution that Shubael Rowly, Jr., should not be further prosecuted on his indictment. As Christophers pointed out, the Assembly seems to have considered the answering of the bond as equivalent to the miscreant's conviction. Christophers was, as the informer, entitled to the reward of £20, which the Court advised him to seek of the Assembly and which that body finally granted him in May, 1713.

Joseph Elderkin

Jonas Hambleton and Paul Wentworth both of New London, and Joseph Elderkin of Norwich were brought before the Superior Court held in New London in September, 1712, on suspicion of having passed an altered Connecticut bill but it was discovered that the first two were not involved and they were cleared by proclamation, each being ordered to pay costs of £5/11/9. Elderkin, however, was indicted for uttering a 2/6 Connecticut bill altered to 20s., to which charge he pleaded not guilty. He was tried, convicted and sentenced to spend six months in prison and to pay a fine of £15 and costs of £10/18/6. Paul Wentworth, who had informed against Elderkin, in open court requested that his costs and those of Hambleton be deducted from the reward of £20 due him as informer, and the court ordered Elderkin to pay £20 to be turned over to Wentworth.

Elderkin, who was in poor health and feared the consequences of spending the winter in what was doubtless an unheated jail, petitioned the Assembly for "the abatement of his imprisonment," and in October it was voted that "if the petitioner shall give bail to any of the judges of the superiour court to render himself to him or any of the said judges upon command at any time within a twelve month after the session of this Court, the time yet to come of his imprisonment, according to the sentence given against him, shall commence when the prison and weather will allow him to be imprisoned without danger of hazarding his health."

Ann Lockwood

When the Superior Court met at Fairfield on September 3, 1717, it had the task of determining who had altered a 2/6 Rhode Island bill to 10/6. Three persons were involved, Lieutenant Richard Higgenbotham, Sergeant Richard Lockwood and Ann Lockwood, the wife of Gersham Lockwood, Jr., of Greenwich. Higgenbotham was cleared by proclamation and it was ordered that the charges of prosecution be paid out of the public treasury. 23 Richard Lockwood gave bail for his appearance in the amount of £100 but did not come into court, sending a note to the effect that he was too ill to attend because of pains in his neck. His bond was declared forfeited, and a scire facias was issued for his appearance before the next sessions in March. At that time he was brought into court, when his case was continued until September. He appeared then but his case was apparently dropped, and there is no further notice of it.

Ann Lockwood was the real culprit. It was revealed that about the beginning of July Mrs. Richard Higgenbotham went from Cos Cob with four pairs of stockings for Mrs. Lockwood to sell in Greenwich. While in Greenwich Mrs. Higgenbotham sold two pairs of the stockings, one to Joseph Knap for Indian corn and another to Mr. Jessup for four shillings. She left the money and the remaining two pairs of stockings with Mrs. Lockwood. A few days later Lieutenant Richard Higgenbotham and his wife went to Ann Lockwood and gave her a 2s. bill and a 2/6 Rhode Island bill. She was to add this to the 4s. she already had from them and was to purchase for them some wool. When Mrs. Lockwood looked at the Rhode Island half crown bill, she remarked that it was a fair opportunity to change the 2 to a 10 because of a vacancy in the paper. At this Mr. Higgenbotham told her not to do so and she said that she would not.

The temptation, however, proved too strong. She altered the bill and paid it out, together with three 2s. bills, to Benjamin Hobby for nine and a quarter pounds of wool. But soon Hobby found that the altered bill would not pass and returned it to her. Ann, thoroughly frightened, on Saturday, July 13, took the altered bill to the Higgenbothams. She told them it was the way the apple tempted Mother Eve and that she would never do such a thing again. She talked with them for about an hour under a green tree, asking them to take back the bill and to stretch the truth by saying that they had the bill of a stranger. If they would do this, she promised them £20 and said they could live at one end of her house and have the use of her cows. Her husband knew of her crime and so did his brother Joseph, who had informed her that he had a good mind to knock her on the head because her husband was like to be ruined by her confounded tricks. Subsequently Gersham Lockwood begged Higgenbotham to burn the bill and to say nothing about Ann's confession.

Eventually Ann was taken into custody by Constable Joshua Reynolds. In September she was indicted for having altered the bill, pleaded not guilty, was tried, convicted and sentenced to stand in the pillory on three several lecture days or days of public meeting for a half hour each day. She was further to be disabled to give any evidence before any court, magistrate, or justice of the peace and was to pay costs of prosecution amounting to £6/13/6. On Saturday, September 7, she was discharged on bail provided by her husband on condition that she would appear at Fairfield on the public days appointed by the Deputy Governor to receive such parts of her punishment as had not yet been executed.

The Oblong Gang and Associates

In 1744 the attention of the authorities in Connecticut, as well as in other provinces, was directed to a band of counterfeiters residing in the Oblong or Equivalent Tract, which had been ceded by Connecticut to New York on May 14, 1731. In a letter dated August 18, 1744, Governor Morris wrote to Governor Clinton of New York about the matter and sent along several examinations and papers concerned with the business. Clinton communicated his information to Governor Jonathan Law of Connecticut, who sent instructions to several justices of the peace to inquire into the matter.

Governor Law on January 2, 1745, wrote from Milford to Governor Clinton:

I have lately received an account from one of our Justices near ye Western Borders of this Gov^t that he has committed one Andrew Nelson to Goal for putting off a Counterfeit 20s. Bill of Rhoad Island equal to 4" wth w^m he found 72" of ye same sort, and the place where this Wickedness is supposed to be carryd on is the Oblong and it is probable that great Quantities of it are handed about by a confederated Gang of w^{ch} I thot fit to advise you...

Nelson, who was, as has been seen, in custody early in January, swore a number of false bills, which were either found on his person or had been traced to him, upon Israel Keith and Samuel Browne of Dover, New York, and Benjamin Stone of Litchfield. Nelson was released on bail provided by himself and his father, William Nelson, for his appearance at the Superior Court in New Haven in August. On August 19, 1745, Justice Samuel Hutchinson issued instructions to the constables to summon as witnesses against Nelson Captain John Sprague, John Gay, James Betts, John Neland and Daniel Parke, all of Sharon. In his indictment Nelson was charged with having on the evening of December 3, 1744, in Sharon, passed a false 20s. Rhode Island bill to James Betts. When the court convened and Nelson was called, he did not appear.

An explanation was forthcoming, for a letter, signed by Andrew Nelson and his father, had been sent to Samuel Darling of New Haven. It stated that Andrew had been pressed into the King's service, had got a substitute and that the substitute had fallen ill. The captain then insisted that Andrew serve. Accompanying documents showed that Captain Leonard Hoar, acting on orders from Colonel John Stoddard, had impressed Andrew to serve in guarding the western frontier and ordered him to impress his father's firelock gun for his use. In this way Nelson escaped almost certain conviction.

Before long more of these 20s. Rhode Island bills were passed by men from the Oblong: Jeremiah Thornton on February 5, 1745, at Colchester passed to James Glass of that town such a forged bill of the emission of 1741. Glass detected the cheat and reported the matter to Nathaniel Foot, J.P., of Colchester, who had Thornton arrested. On the same day Thomas Cooper, also from the Oblong, uttered to Joseph Chamberlain in Colchester another counterfeit Rhode Island bill. Both men were tried and convicted at the March session of the Superior Court in Hartford and were sentenced in accordance with law. On May 9, 1745, these two criminals, encouraged, no doubt, by previous action of the Assembly in similar cases, petitioned for release from life imprisonment in case they could find someone to pay their expenses and charges. Their prayer was granted on condition that they pay all costs and charges and £20 each (the rewards given to the informer or informers against them, one of whom was James Glass) and with the understanding that if they were ever found in the colony after the ten days following their release had elapsed they were to be returned to the workhouse for life.

In addition to these two members of the Oblong gang still another two, Joseph Boyce, Sr., and John Scias (also spelled Scious and Syas) had been taken up, through the efforts of Robert Clark of Uxbridge, Massachusetts, and lodged in the jail in Hartford. In May Clark requested of and received from the Assembly aid in having the two offenders transported to Hampshire County in Massachusetts.

Other members of the gang appear to have been Joseph Boyce, Jr., Samuel Thompson, Joseph Plummer, Henry Bosworth, Israel Keith of New Sherburn, Seth Sherwood and a certain Hurlburt. It also seems likely that Justice Daniel Hunt and Captain Augustine Hunt were somehow involved. Some of these persons were apprehended, as is shown by the following letter of June 19, 1745, sent by Governor Law to Governor Shirley. Law wrote:

Saturday night was Sennit a Justice of peace on our western Borders informed me of one who Contrived to Expose young Boyce and others to be taken in ye Very act of using ye Counterfeit plates in a Certain Swamp in ye Oblong on tuesday following but it being out of this Gover^{mt} I sent ye Justice directly to Gov^r Clinton to Inform of ye Stratagem thinking nothing was wanting but an authority & assistance Sufficient would readily be had of our people within ten miles of ye Spot, he Shewed me two rhoad island xx^s bills one with Divers mistakes in it ye other with these errors rectified taken of ye day before, and ye Justice returned with a Letter ye Gov^r Signifying y^t ye Council were of opinion yt yr was no foundation for a warrant, ye Justice being able to Sware only to here Says but ye undertaker had found ye plates a 20 s Rh and a half a Crown Plate & a N.Y. plate of 20^s not perfectly Compleated, Press cloths and other implements &c: Sends them over ye line, Decoys Boyce & one Hurlburt a partner into ye Edge of this Gov^{mt} Seizeth them & they are in N. Haven Goal Hurlburt Confesseth himself Guilty and accuseth 22 persons as Confederate with them Boyces father and Scious were transported through this Gov^{mt} to you some time Since.

The persons concerned in giving information against or seizing these malefactors (Sherwood, Boyce, Nelson and Hurlburt) were William Drinkwater, who informed against Sherwood, James Betts, who informed against Andrew Nelson, and William Spencer and Ephraim Seeley. The Connecticut Assembly voted Drinkwater and Betts £20 each, while Seeley was given £50 for having helped to detect the criminals and because it was feared he might suffer from the vengeful practices of the delinquents and their associates. Spencer, aided by others, had probably taken an active part in the capture of some of the counterfeiters, all of whom escaped conviction, since some were released on bail, which they forfeited, and others escaped from jail. The two who broke jail were Hurlburt and Joseph Boyce, who escaped from prison in New Haven between July 18 and August 21, leaving only their plates in the hands of the authorities. Sherwood, like Nelson, must have been released on bail and forfeited his bond by failing to appear.

Joseph Holmes

Joseph Holmes of Hatfield in Hampshire County, Massachusetts, was indicted at the Superior Court in Hartford on September 1, 1761, for having on August 29 at Middletown passed off five counterfeit Spanish milled dollars, one to Samuel Starr, one to Thomas Danforth, one to Matthew Talcott and two to Abigail Shayler. He pleaded guilty and was sentenced to have his right ear cut off, to be given twenty-five lashes on the naked body and to pay costs of £14/9/9.

Jonathan Olds

On January 27, 1763, Samuel Pettibone, King's Attorney of Litchfield County, complained to Justice John Patterson against Jonathan Olds of Egrimont Parish in Sheffield, Berkshire County, Massachusetts. He charged that at Cornwall on January 25 Olds made thirty Spanish dollars and the following day passed one of them to Hopestill Pierce, wife of Lieutenant Joshua Pierce of Cornwall, another to the wife of Jeremiah Griswold of Litchfield and a third to some person in Sharon. John Pierce, Constable of Cornwall, arrested Olds on January 27 and the prisoner was examined and bound over in bail of £100 to the August term of the Superior Court in Litchfield. Olds pleaded guilty and was sentenced to be whipped thirty lashes and to pay costs of £20/8-. Sheriff Oliver Wolcott had him whipped at the sign post in Litchfield and then sent back to jail, doubtless because the costs were not paid.

James Sturdevant

A complaint was made on February 23, 1770, to Justice Michael Humphry that Jesse and George Tobey and James Sturdevant of Norfolk had coining instruments and were making coin. A warrant was issued and Constable Josiah Starr of New Milford apprehended Sturdevant, upon whom he found a recipe containing in fixed proportions arsenic, sublimate, sal ammoniac, salt of tartar, borax and potash, evidently to be used in coining. Sturdevant was bound over to the Superior Court to be held in Litchfield and was then released on bail of £100, furnished by Caleb Knap, Nathan Sturdevant and Jonathan Pinney, all of Norfolk. On March 19 Justice Humphry issued a search warrant but Constable Giles Pettibone could find neither coining instruments nor metal. At the Superior Court Sturdevant was indicted for having on July 20, 1769, counterfeited several Spanish dollars and pistareens and for having on October 20 in Norfolk passed one of the false dollars to Samuel Knap, Jr., of that town. He was tried, convicted and sentenced to pay a fine of £50 and costs. Apparently Jesse and George Tobey were not arrested, or, if they were, they were not bound over to the Superior Court.

Timothy Keys

The grand jurors of Norfolk on September 15, 1770, informed Justice Michael Humphry that Timothy Keys of New Marlborough in Berkshire County, Massachusetts, had in Norfolk an instrument for coining dollars and that he had made dollars and other coins. A warrant was issued for the arrest of the suspected coiner but it was found that he had fled to Massachusetts. Justice John Ashley in Berkshire County also issued a writ for Keys's arrest and the Sheriff of Berkshire County pursued Keys across the line to Norfolk, where on September 17 the fugitive was captured by John Phelps. It was charged that on March 31, 1769, at Norfolk Keys had made ten false dollars and passed one of them to Reuben Stevens of Canaan. When the case came up in court, Keys pleaded that the facts alleged against him were done more than a year before the commencement of the suit and hence were barred by the statute of limitations. It was decided by the court that the plea in abatement was sufficient

Coiners in Colchester

Daniel Isham on March 21, 1771, complained to Justice Daniel Groot that John Newton, Jr., of Colchester had made Spanish dollars and gold coin. Newton was at once arrested and at his examination on the following day admitted that he had made molds, had cast pewter dollars in sand and had then hidden the coins in his shop. He had, he confessed, showed two of the dollar molds at his shop to Asahel Newton and Joseph Chapman. At New London, he said, he had passed a false pistareen or an English shilling to Captain Douglas, who had refused to accept it. He was bound over to the Superior Court to be held in Hartford in September and then released on bail of £200 furnished by himself and by Israel Newton of Colchester. At the Superior Court he was indicted for having about March 5 at Colchester constructed a mold for making dollars and for having cast about forty coins with it. Despite his admissions to Justice Groot he pleaded not guilty, was tried, acquitted and dismissed on payment of costs.

Isham likewise complained that Joseph Chapman, who had previously lived at Great Barrington but was then residing at Colchester, had at some time after September 1, 1770, stamped dollars and passed some of them. Justice Groot issued a writ for Chapman's arrest and he was taken up on March 21 by Constable Elihu Clark of Colchester. At his examination before the magistrate he, too, talked freely and incriminated Asahel Newton. He and Asahel, he stated, had secured two molds made by John Newton in his shop and they paid John twelve shillings for one of them and borrowed the other. One evening at Asahel's house in Colchester he (Chapman) and Asahel ran seven dollars out of pewter, of which he (Chapman) passed one to James Morgan, who later returned it. According to Chapman, Asahel had in a chest a mold and twenty or twenty-five counterfeit dollars. Chapman was bound over to the Superior Court and released on bail of £100, provided by himself and Joseph Tubbs of Colchester, for his appearance in court in September. He failed, however, to appear and his bond was declared forfeited.

appendix 2 - SuperCollider code and Lilypond template

cicc_readme.scd

```
1  /*
2  ----execute
3  Execute cicc_main.scd to run.
4
5
6  ----transport tab
7  The play button will always start from the beginning of the current section.
8
9  The transport buttons allow you to advance by subsection (<,>) and section (<<,>>).
10
11 Tempo change will only go into effect once "set tempo" button is pressed
12
13 Turning the "auto advance" button on will automatically move from one subsection to the other.
14
15 Setting the address:port will create a pipe to receive a message to advance the subsection externally with an OSC message '/nextSubsection. This could be used to set up a
   foot pedal / controller for the guitarist to advance the subsections manually.
16
17 Turning the "interludes" button on will automatically fade in the interlude synth at the start of the ultimate subsection of each section. This will turn off automatic
   advance between the ultimate subsection of a section and the first subsection of the following section. That is, the performer will have manually advance after
   the last subsection of each section. The interlude synth will automatically fade out once a new section has been triggered.
18
19 Set order takes comma and dash delimited values; e.g.: "1, 2, 3, 4 - 10" will play from section 1 to section 10 and "5 - 10, 1, 2, 3" will play from section 5 to section
   10 and then from section 1 to section 3. This will only go into effect once the "set order" button is pressed.
20
21 The default seed given in the application and reseeded when the "reset seed" button is pressed will generate the default music and score (as provided). Changing the seed
   will generate a new version with that seed once the "generate" button is pressed. After the new version is generated, new Lilypond files can be generated by
   pressing the "transcribe" button. This will create a cicc.score.ly file in a folder labeled "seed.[number]" which can be rendered by Lilypond. Note that the file
   must be rendered from that location as it depends in files in that folder and the "includes" subfolder.
22
23
24 ----mixer tab
25 This allow individual control of each of the sonic elements. The synthesized guitar part is automatically muted is at should only be used for audition and practice. The low
   accompaniment has two separate tracks in case a performer cannot play both the notes. The sonification will go to outputs 1 and 2 while the click will go to
   outputs 3 and 4.
26 */
```

cicc_main.scd

```
1  (
2  // MAIN LAUNCH (loads necessary files and definitions)
3
4  var appEnvironment;
5
6  //push new environment
7  appEnvironment = Environment.make;
8  appEnvironment.push;
9
10 s.waitForBoot({
11
12     `hash = Date.getDate.hash.asString;
13
14     // load all files
15     "cicc.musical_data_generator.scd".loadRelative;
16     "cicc.sonifier.scd".loadRelative;
17     "cicc.gui.scd".loadRelative;
18     "cicc.transcriber.scd".loadRelative;
19
20     // generate all the data
21     `genAll = {arg seed;
22         `allMusicData = `genMusicData.value(seed);
23         `patterns = `allMusicData[0];
24         `scoreData = `allMusicData[1];
25         `sectionOffsets = `allMusicData[2];
26         `currentSection = 0;
27         `currentSubsection = 0;
28         `isPlaying = false;
29     };
30
31     // set the global variables
32     `tempoClock = TempoClock.new(90 / 60);
33     `dir = thisProcess.nowExecutingPath.dirname;
34     "loading app".postln;
35     `genAll.value(20200525);
36     `play = Synth.new(`masterPlayerControl ++ `hash);
37     {
38         var center, interval, freq1, freq2, tremRate;
39         center = 50 - 12.0.rand;
40         interval = 3.0.rand + 2;
41         freq1 = (center + (interval / 2)).midicps;
42         freq2 = (center - (interval / 2)).midicps;
43         tremRate = 50 + 4.0.rand2;
44         `interludeTremelo = Synth.new(`interludeTremelo ++ `hash, [\freq1, freq1, \freq2, freq2, \tremRate, tremRate]);
45     }.value;
46     `autoAdvance = true;
47     `interludes = false;
48     `sectionOrder = `patterns.size.collect({arg sec; sec});
49     `generateGUI.value;
50     "ready".postln;
51 });
52 appEnvironment.pop;
53 )
```

cicc_musical_data_generator.scd

```
1  (
2  var genInitSeq, finalizeSeqs, finalizeAccompHigh, finalizeAccompLow;
3
4  //----init vars for initial sequence generation
5  genInitSeq = {arg seed = 20200525;
6      var setDur, strings, stringIndex, state, lastStrings, position, dur, openStringCount, landingCount, sectionCount,
7      modelInitSeq, res;
8
9      thisThread.randSeed = seed;
10
11      //----helper dur function
12      setDur = {arg probs; [2, 3, 4, 5.rand + 3].wchoose(probs.normalizeSum)};
13
14      modelInitSeq = 16.collect({
15          [
16              //probably adjustment for altering picking pattern
17              2 + 1.0.rand2,
18              //probabilities adjustment for position
19              7 + 2.0.rand2,
20              //probabilities for inserting walk down
21              2.5 + 1.5.rand2,
```

```

22 //penultimate position
23 6.collect({2.rand + 1}),
24 //probabilities for adjustment to duration
25 3 + 1.0.rand2,
26 //probabilities for note durations
27 [5 + 2.0.rand2, 5 + 2.0.rand2, 5 + 2.0.rand2, 1],
28 //probabilities adjustment for altering state
29 2 + 1.0.rand2,
30 //number of notes in ultimate section
31 12 + 4.rand
32 });
33
34 strings = (0..5);
35 state = 6.collect({[0, 1].wchoose([2, 1].normalizeSum)}); //fretted or not
36 lastStrings = [nil, nil];
37 position = 6.collect({10 + 3.rand}); //which frets
38 dur = setDur.value(modelInitSeq[0][5]);
39 openStringCount = 0;
40 landingCount = 0; //for extending section landing on open strings
41 sectionCount = 0;
42
43 res = []; //notes before the more static repetitions are put in
44
45 //----run routine and create template sequence
46 //----number of sections must be even - generate 16 by default
47 ({sectionCount < 16}).while({
48   var alterPattern, penultimatePos, lastFrettedString, forceUltimateDescent;
49
50   //alter string pattern or not
51   penultimatePos = (position.sign.sum == 1);
52   if(penultimatePos, {lastFrettedString = position.sign.indexOf(1)});
53   forceUltimateDescent = penultimatePos && strings.includes(lastFrettedString).not;
54   alterPattern = [true, forceUltimateDescent].wchoose([1, modelInitSeq[sectionCount][0]].normalizeSum);
55   if(alterPattern, {
56     //var lastFrettedString;
57     strings = (0..5).scramble[. (4.rand + 1)];
58     //keep selecting until you have the final string
59     while({forceUltimateDescent && strings.includes(lastFrettedString).not}, {
60       strings = (0..5).scramble[. (4.rand + 1)];
61     });
62     //rotate if a note gets repeated
63     if(lastStrings.last == strings.first, {strings = strings.rotate});
64     lastStrings = strings;
65   });
66
67   //iterate through the strings
68   strings.do({arg string, stringIndex;
69     var alterPos;
70
71     //alter fret if fretted and keeping hand in similar position
72     alterPos = (position[string] * state[string]) > 0;
73     alterPos = alterPos && (state[string] == 1); //isFretted
74     alterPos = alterPos && (position[string] > (position.maxItem - 3));
75     alterPos = [alterPos, false].wchoose([modelInitSeq[sectionCount][1], 1].normalizeSum);
76     if(alterPos, {
77       var walkDown, stepLimit;
78
79       //walk down or not
80       walkDown = [true, false].wchoose([1, modelInitSeq[sectionCount][2]].normalizeSum);
81       if(walkDown, {
82         res = res.add([string, state[string] * position[string], dur, position.deepCopy]);
83       });
84
85       //make sure a hand position is not too wide
86       stepLimit = (position.maxItem - (position[string] - 2)) != 4;
87       if(stepLimit.not, {
88         position[string] = (position[string] - 1).clip(0, 12);
89       });
90       if(stepLimit, {
91         position[string] = (position[string] - [1, 2].choose).clip(0, 12);
92       });
93     });
94   }, {
95     if((position[string] <= modelInitSeq[sectionCount][3][string]) && (state[string] == 0), {position[string] = 0});
96   });
97
98   //alter duration or not
99   if([true, false].wchoose([modelInitSeq[sectionCount][4], 1].normalizeSum), {dur = setDur.value(modelInitSeq[sectionCount][5])});
100
101   //add
102   res = res.add([string, state[string] * position[string], dur, position.deepCopy]);
103
104   //alter state or not favoring off if on (determines if string is open or fretted)
105   if(sectionCount.even, {
106     var isFretted, probs, alterState;
107     isFretted = (state[string] == 1);
108     probs = [if(isFretted, {modelInitSeq[sectionCount][6]}, {1}), 1].normalizeSum;
109     alterState = [true, false].wchoose(probs);
110     if(alterState, {state[string] = (state[string] + 1) % 2});
111   });
112
113   //alternate option
114   if(sectionCount.odd, {
115     var isFretted, alterable;
116     isFretted = (state[string] == 1);
117     alterable = isFretted || ((state[string] == 0) && (state.sum < 3));
118     if(alterable, {
119       var probs, alterState;
120       probs = [if(isFretted, {1}, {modelInitSeq[sectionCount][6]}), 1].normalizeSum;
121       alterState = [true, false].wchoose(probs);
122       if(alterState, {state[string] = (state[string] + 1) % 2});
123     });
124   });
125
126   //reset if everything arrives at the bottom string
127   if(position == [0, 0, 0, 0, 0, 0], {
128     var noNotes, hasLanded;
129     noNotes = modelInitSeq[sectionCount][7];
130     hasLanded = (landingCount > noNotes) && (stringIndex == (strings.size - 1));
131     if(hasLanded, {
132       (landingCount - 1).do({arg index;
133         res[res.size - index - 1][2] = (dur * (1 + ((1 - (index / landingCount).clip(0, 1).pow(0.5)) * 8))).asInteger
134       });
135       position = 6.collect({10 + 3.rand});
136       landingCount = 0;
137       sectionCount = sectionCount + 1;
138     });
139     if(hasLanded.not, {landingCount = landingCount + 1})
140   });
141 })
142 };
143 res
144 };
145
146

```

```

147 //-----insert more static sections by repeating a figure
148 finalizeSeqs = {arg initSeq;
149   var modelReps, extendToBeat, insertTS,
150   timeStampSection, timeStampTotal, timeStampSectionStart, lastDur, lastPos,
151   sectionSeq, timeSigInsSeq, state, sectionCount, guitarSeq;
152
153   modelReps = 16.collect({
154     [
155       //where in the descent will the repetitions occur
156       4 + 4.rand,
157       //length of repetition
158       15.rand + 5,
159       //number of repetitions
160       10.rand + 5,
161       //probabilities for keeping a note in the repetition
162       5 + 1.0.rand2,
163       //max interval of bass part in repetitions
164       5 + 3.rand
165     ]
166   });
167
168   extendToBeat = {arg seq, round = 4;
169     var timeStampTotal, altEndDur;
170     //this makes sure it is some multiple of a beat
171     timeStampTotal = seq.slice(nil, 2).sum;
172     altEndDur = timeStampTotal.round(round) - timeStampTotal;
173     //must remain larger than a 16th notes
174     if((seq.last[2] + altEndDur) <= 1, {altEndDur = altEndDur + round});
175     seq.last[2] = seq.last[2] + altEndDur;
176     [seq, altEndDur];
177   };
178
179   insertTS = {arg seq, timeStampSectionStart, type, accompSwitch;
180     var timeStampTotal, noMeasures;
181     timeStampTotal = seq.slice(nil, 2).sum;
182     sectionSeq = sectionSeq.add([timeStampTotal, type, accompSwitch]);
183     noMeasures = ((timeStampTotal - timeStampSectionStart) / 16);
184     if(noMeasures.frac > 0, {
185       timeSigInsSeq = timeSigInsSeq.add(
186         // make 3/2 instead of 1/2
187         if((noMeasures.frac / 0.25).asInteger != 2, {
188           [timeStampTotal - (4 * (noMeasures.frac / 0.25).asInteger), (noMeasures.frac / 0.25).asInteger]
189         }, {
190           [timeStampTotal - (4 * 6), 6]
191         }
192       );
193       timeSigInsSeq = timeSigInsSeq.add([timeStampTotal, 4]);
194     });
195   };
196
197   timeStampSection = 0; //track time in each section
198   timeStampTotal = 0; //track overall time
199   timeStampSectionStart = 0; //track the time of the start of a section
200   lastDur = initSeq[0][2]; //helper for time signature data
201   lastPos = initSeq[0].last; //helper for keeping track of landing.
202
203   guitarSeq = []; //this is the final sequence with repetitions inserted
204   sectionSeq = [[0, 0, true]]; //sequence of times for each section (used for double bars in score)
205   timeSigInsSeq = [[0, 4]]; //sequence for insertion of time signatures and double bars;
206
207   state = 0;
208   sectionCount = 0;
209
210   initSeq.do({arg item, index, altEndDur;
211     var dur, pos;
212     dur = item[2];
213     pos = item.last;
214
215     if(state != 1, {
216       // basically this just copies the original template over
217       var landingBorder, sectionBorder;
218       landingBorder = (pos == [0, 0, 0, 0, 0, 0]) && (lastPos != [0, 0, 0, 0, 0, 0]);
219       sectionBorder = (pos != [0, 0, 0, 0, 0, 0]) && (lastPos == [0, 0, 0, 0, 0, 0]);
220       if(landingBorder || sectionBorder, {
221         var seqExtPair;
222         seqExtPair = extendToBeat.value(guitarSeq, 8);
223         guitarSeq = seqExtPair[0];
224         timeStampSection = timeStampSection + seqExtPair[1];
225         insertTS.value(guitarSeq, timeStampSectionStart, if(pos == [0, 0, 0, 0, 0, 0], {1}, {-1}), false);
226         timeStampSectionStart = guitarSeq.slice(nil, 2).sum;
227       });
228
229       if(sectionBorder, {
230         state = 0;
231         sectionCount = sectionCount + 1;
232       });
233
234       guitarSeq = guitarSeq.add(item.add(-1));
235       timeStampSection = timeStampSection + dur;
236
237       lastDur = dur;
238       lastPos = pos;
239
240       if((state == 0) && (pos.minItem < modelReps[sectionCount][0]), {state = 1});
241     });
242
243     if(state == 1, {
244       // grabs a figure and repeats it altering it subtly
245       var rec, reps, noMeasures;
246
247       guitarSeq = extendToBeat.value(guitarSeq, 8)[0];
248       timeStampTotal = guitarSeq.slice(nil, 2).sum;
249       insertTS.value(guitarSeq, timeStampSectionStart, 0, true);
250       timeStampSectionStart = timeStampTotal;
251       rec = guitarSeq[guitarSeq.size - modelReps[sectionCount][1]..guitarSeq.size].deepCopy;
252       reps = modelReps[sectionCount][2];
253       reps.do({arg index;
254         rec.do({arg item, rIndex;
255           var add, dur;
256           add = if(index == 0, {3}, {0});
257           dur = (item[2] + 2.rand2 + add);
258           if(dur < 2, {dur = [0, 2].wchoose([1, 4].normalizeSum)});
259           rec[rIndex] = [item[0], item[1], dur];
260           if([true, false].wchoose([modelReps[sectionCount][3], 1].normalizeSum), {
261             guitarSeq = guitarSeq.add(rec[rIndex].add(modelReps[sectionCount][4] * (1 - ((1 / reps) * index))));
262           });
263
264           // if chord randomly choose one of the notes
265           if(guitarSeq.last[2] == 0, {
266             arg toAdd = [];
267             toAdd = toAdd.add(guitarSeq.pop);
268             toAdd = toAdd.add(guitarSeq.pop);
269             toAdd[0][2] = toAdd[1][2];
270             toAdd[1][3] = toAdd[0][3];
271             toAdd = toAdd.choose;

```

```

272         guitarSeq = guitarSeq.add(toAdd);
273     };
274 };
275
276 if(index < (reps - 1), {
277     guitarSeq = extendToBeat.value(guitarSeq, 4)[0];
278 }, {
279     guitarSeq = extendToBeat.value(guitarSeq, 8)[0];
280 });
281
282 });
283
284 insertTS.value(guitarSeq, timeStampSectionStart, 0, true);
285
286 timeStampSection = 0;
287 timeStampSectionStart = guitarSeq.slice(nil, 2).sum;
288 lastDur = initSeq[index + 1][2];
289 state = 2;
290 });
291 };
292 [guitarSeq, sectionSeq, timeSigInsSeq]
293 };
294
295 // add the high note part
296 finalizeAccompHigh = {arg sectionSeq;
297     var accompHighSeq, timeStamp, subSecType, modelAccomp;
298     accompHighSeq = [];
299     timeStamp = 0;
300     subSecType = 0;
301
302     modelAccomp = sectionSeq.size.collect({
303         [
304             //short probability
305             1.5 + 0.5.rand2,
306             //rest probability
307             3 + 1.0.rand2,
308             //short note average
309             20 + 5.rand2,
310             //short note range
311             5 + 3.rand2,
312             //long note average
313             50 + 10.rand2,
314             //long note range
315             10 + 5.rand2,
316             //rest average
317             40 + 10.rand2,
318             //rest range
319             5 + 5.rand2,
320             //internote space (short rest)
321             6.rand
322         ]
323     });
324
325 sectionSeq.do({arg subSecData, subSecIndex;
326     var subSecEnd, freq, noRestCount, shortCount;
327     subSecEnd = subSecData[0];
328     freq = if(subSecIndex.even, {62.midicps * 8}, {62.midicps * 8 * 6/5});
329     if(subSecData.last, {subSecType = ((subSecType + 1) % 2)});
330     noRestCount = 0;
331     shortCount = 0;
332     while((timeStamp < subSecEnd), {
333         var dur, sus, isShort, insertRest;
334
335         isShort = case
336         {shortCount == 0} {true}
337         {shortCount < 3} {[true, false].wchoose([modelAccomp[subSecIndex][0], 1].normalizeSum)}
338         {true} {false};
339
340         insertRest = [true, noRestCount > 3].wchoose([modelAccomp[subSecIndex][1], 1].normalizeSum);
341
342         if(isShort, {
343             sus = (modelAccomp[subSecIndex][2] + modelAccomp[subSecIndex][3].rand2).round(2);
344             shortCount = shortCount + 1;
345         }, {
346             sus = (modelAccomp[subSecIndex][4] + modelAccomp[subSecIndex][5].rand2).round(2);
347             shortCount = 0;
348         });
349
350         if(insertRest, {
351             dur = sus + (modelAccomp[subSecIndex][6] + modelAccomp[subSecIndex][7].rand2).round(2);
352             noRestCount = 0;
353         }, {
354             dur = sus + 2 + modelAccomp[subSecIndex][8].rand.round(2);
355             noRestCount = noRestCount + 1;
356         });
357
358         if((timeStamp + dur) < subSecEnd, {
359             accompHighSeq = accompHighSeq.add([freq, dur, sus.clip(0, dur)]);
360         }, {
361             var remainder;
362             remainder = (subSecEnd - timeStamp);
363             sus = if(remainder > 10, {(remainder - 10).rand + 8}.round(2), {0});
364             dur = (remainder + 10.rand).clip(2, 1000).round(2);
365             accompHighSeq = accompHighSeq.add([freq, dur, sus]);
366         });
367         timeStamp = timeStamp + dur;
368     });
369 };
370 accompHighSeq
371 };
372
373
374 // add the low note part
375 finalizeAccompLow = {arg guitarSeq, sectionSeq;
376     var accompLowSeq, durAccum, lastTrigVal;
377     accompLowSeq = [];
378     durAccum = 0;
379     lastTrigVal = 0;
380     guitarSeq.do({arg item, i;
381         var dur, trig, freq1, freq2, finalDur;
382         dur = item[2];
383         trig = item.last;
384         if(lastTrigVal != trig, {
385             freq1 = if(trig > -1, {62.midicps / 4 * 3/4}, {62.midicps / 4});
386             freq2 = freq1 + if(trig > -1, {trig}, {0});
387             finalDur = durAccum;
388             accompLowSeq = accompLowSeq.add([freq1, freq2, finalDur]);
389             durAccum = 0;
390         });
391         durAccum = durAccum + dur;
392         lastTrigVal = trig;
393     });
394
395     accompLowSeq = [accompLowSeq.slice(nil, 0), accompLowSeq.slice(nil, 1), accompLowSeq.slice(nil, 2).integrate].flop;
396     sectionSeq.collect({arg section, secIndex;

```



```

397     if(section[1] == 1, {
398         var curTime, secLength;
399         curTime = section[0];
400         secLength = section[0] - sectionSeq[secIndex - 1][0];
401         accompLowSeq = accompLowSeq.add([62.midicps / 8, (62.midicps / 8) + 0, curTime]);
402         curTime = curTime - (50.rand + 50).clip(0, (secLength / 3) - 5).round(4).asInteger;
403         accompLowSeq = accompLowSeq.add([64.midicps / 8, (64.midicps / 8) + 2 + 1.0.rand2, curTime]);
404         curTime = curTime - (50.rand + 50).clip(0, (secLength / 3) - 5).round(4).asInteger;
405         accompLowSeq = accompLowSeq.add([65.midicps / 8, (65.midicps / 8) + 4 + 1.0.rand2, curTime]);
406     });
407     if(section[1] == -1, {
408         var curTime = section[0];
409         accompLowSeq = accompLowSeq.add([62.midicps / 4, (62.midicps / 4) + 0, curTime]);
410     });
411 };
412
413 accompLowSeq = accompLowSeq.sort({ arg a, b; a[2] < b[2] });
414 accompLowSeq = [accompLowSeq.slice(nil, 0), accompLowSeq.slice(nil, 1),
415     accompLowSeq.slice(nil, 2).differentiate.drop(1).add(1)].flop;
416
417 accompLowSeq
418 };
419
420 `genMusicData = {arg seed;
421     var initSeq, finalSeqs, guitarSeq, accompHighSeq, accompLowSeq, sectionSeq, timeSigSeq,
422     patterns, scoreData, sectionOffsets;
423
424     initSeq = genInitSeq.value(seed);
425     finalSeqs = finalizeSeqs.value(initSeq);
426     guitarSeq = finalSeqs[0];
427     accompHighSeq = finalizeAccompHigh.value(finalSeqs[1].deepCopy.add([finalSeqs[0].slice(nil, 2).sum, -1, false]));
428     accompLowSeq = finalizeAccompLow.value(finalSeqs[0], finalSeqs[1]);
429     sectionSeq = finalSeqs[1];
430     timeSigSeq = finalSeqs[2];
431
432     patterns = `genPatterns.value(guitarSeq, accompLowSeq, accompHighSeq, sectionSeq);
433     scoreData = `genScoreData.value(guitarSeq, accompLowSeq, accompHighSeq, timeSigSeq, sectionSeq);
434     sectionOffsets = sectionSeq.slice(nil, 0);
435
436     [patterns, scoreData, sectionOffsets]
437 };
438 )

```

cicc_sonifer.scd

```

1  (
2  //busses
3  `masterBus = Bus.audio(s, 1);
4  `guitarBus = Bus.audio(s, 1);
5  `accompHighBus = Bus.audio(s, 1);
6  `accompLowLowerBusA = Bus.audio(s, 1);
7  `accompLowUpperBusA = Bus.audio(s, 1);
8  `accompLowLowerBusB = Bus.audio(s, 1);
9  `accompLowUpperBusB = Bus.audio(s, 1);
10 `interludeTremoloBus = Bus.audio(s, 1);
11 `clickBus = Bus.audio(s, 1);
12
13 SynthDef(`masterPlayerControl ++ `hash, {
14     arg sel = 0,
15     masterVol = 1, masterMute = 1,
16     guitarVol = 1, guitarPan = 0, guitarMute = 0,
17     accompHighVol = 1, accompHighPan = 0, accompHighMute = 1,
18     accompLowLowerVol = 1, accompLowLowerPan = 0, accompLowLowerMute = 1,
19     accompLowUpperVol = 1, accompLowUpperPan = 0, accompLowUpperMute = 1,
20     interludeVol = 1, interludePan = 0, interludeMute = 1,
21     clickVol = 1, clickPan = 0, clickMute = 1;
22     var guitarSig, accompHighSig, accompLowLowerSig, accompLowUpperSig, interludeSig, clickSig,
23     guitarSigPanned, accompHighSigPanned, accompLowLowerSigPanned, accompLowUpperSigPanned, interludeSigPanned, clickSigPanned,
24     masterSig, imp;
25
26     guitarSig = In.ar(`guitarBus) * guitarVol;
27     accompHighSig = In.ar(`accompHighBus) * accompHighVol;
28     accompLowLowerSig = Mix.ar(
29         [
30             In.ar(`accompLowLowerBusA) * EnvGen.kr(Env.asr(0.001, 1, 0.1), (sel + 1) % 2),
31             In.ar(`accompLowLowerBusB) * EnvGen.kr(Env.asr(0.001, 1, 0.1), sel)
32         ]
33     ) * accompLowLowerVol;
34     accompLowUpperSig = Mix.ar(
35         [
36             In.ar(`accompLowUpperBusA) * EnvGen.kr(Env.asr(0.001, 1, 0.1), (sel + 1) % 2),
37             In.ar(`accompLowUpperBusB) * EnvGen.kr(Env.asr(0.001, 1, 0.1), sel)
38         ]
39     ) * accompLowUpperVol;
40     interludeSig = In.ar(`interludeTremoloBus) * interludeVol;
41     clickSig = In.ar(`clickBus) * clickVol;
42
43     guitarSigPanned = Pan2.ar(guitarSig * guitarMute, guitarPan);
44     accompHighSigPanned = Pan2.ar(accompHighSig * accompHighMute, accompHighPan);
45     accompLowLowerSigPanned = Pan2.ar(accompLowLowerSig * accompLowLowerMute, accompLowLowerPan);
46     accompLowUpperSigPanned = Pan2.ar(accompLowUpperSig * accompLowUpperMute, accompLowUpperPan);
47     interludeSigPanned = Pan2.ar(interludeSig * interludeMute, interludePan);
48     clickSigPanned = Pan2.ar(clickSig * clickMute, clickPan);
49
50     masterSig = Mix.ar(
51         [
52             guitarSigPanned,
53             accompHighSigPanned,
54             accompLowLowerSigPanned,
55             accompLowUpperSigPanned,
56             interludeSigPanned
57         ]) * masterVol * masterMute;
58
59     Out.ar(0, masterSig);
60     Out.ar(2, clickSigPanned); //change this if you want the click to go somewhere else
61
62     imp = Impulse.kr(10);
63     SendReply.kr(imp,
64         '/masterLevels' ++ `hash,
65         values: [Amplitude.kr(masterSig)]);
66     SendReply.kr(imp,
67         '/trackLevels' ++ `hash,
68         values:
69         [
70             Amplitude.kr(guitarSig), Amplitude.kr(accompHighSig),
71             Amplitude.kr(accompLowLowerSig), Amplitude.kr(accompLowUpperSig),
72             Amplitude.kr(interludeSig), Amplitude.kr(clickSig)
73         ]
74     );
75 }) .add;
76
77 SynthDef(`transport ++ `hash, {arg measure = 0, beat = 0, gate = 1, dur = 1;

```



```

79   SendReply.kr(impulse.kr(0) * (measure > 0) * (beat > 0), '/measureClock' ++ `hash, values: [measure, beat]);
80   SendReply.kr(impulse.kr(0) * (measure < 1) * (beat < 1), '/nextSubsection' ++ `hash);
81   EnvGen.kr(Env.sine(dur), gate, doneAction: 2);
82   }).add;
83
84
85   SynthDef(\click ++ `hash, {arg beat = 0, gate = 1, dur = 1;
86     Out.ar("clickBus", 10 * BPF.ar(WhiteNoise.ar * EnvGen.kr(Env.perc(0.01, 0.1), gate), 440 * ((beat <= 1) + 1), 0.02));
87     EnvGen.kr(Env.sine(dur), gate, doneAction: 2);
88   }).add;
89
90
91   //-----karplus
92   SynthDef(\karplus ++ `hash, {arg freq, gate = 1, amp = 0.5, bus;
93     Out.ar(bus,
94       Pluck.ar(WhiteNoise.ar(0.1), Impulse.kr(0), 220.reciprocal, freq.reciprocal, 10, coef:0) *
95       Linen.kr(gate, doneAction: 2) * amp)
96   }).add;
97
98
99   //-----accompaniment
100  SynthDef(\accompBass ++ `hash, {arg freq1 = 100, freq2 = 100, gate = 1, amp = 0.5, busLower, busUpper, cutoff = 0;
101    var env, lower, upper;
102    env = EnvGen.kr(Env.perc(0.1, 10, level: amp), Impulse.kr(0) + Changed.kr(freq2));
103    lower = SinOsc.ar(freq1, 0, 0.5) * env;
104    upper = SinOsc.ar(freq2, 0, 0.5) * env;
105    Out.ar(busLower, lower);
106    Out.ar(busUpper, upper)
107  }).add;
108
109
110  //this is not releasing properly
111  SynthDef(\accompTreble ++ `hash, {arg freq, gate = 1, sustain, amp, bus;
112    var treble;
113    treble = SinOsc.ar(freq, 0, EnvGen.kr(Env.linen(0.3, 0, 0.7, amp * 0.075, \sine), gate, timeScale: sustain, doneAction: 2));
114    Out.ar(bus, treble)
115  }).add;
116
117  //-----interlude
118  //note that this is sensitive to frequency and tremolo rate inputs
119  SynthDef(\interludeTremolo ++ `hash, {arg gate = 0, amp = 1, freq1, freq2, tremRate;
120    var tremoloTrig, trem, freq, sig, feedback, fade;
121    //fast tremolo - note that this can be slower so long as the delaytime of the feedback remains short
122    tremoloTrig = Impulse.kr(tremRate);
123    //tremolo between two notes
124    trem = Select.kr(Stepper.kr(tremoloTrig, 0, 0, 1), [freq1, freq2]);
125    //occasionally tremolo on same note
126    freq = Select.kr(TWChoose.kr(Dust.kr(10), [0, 1, 2], [5, 1, 1], 1), [trem, freq1, freq2]);
127    //generate signal
128    sig = VarSaw.ar(freq, 0, 0.3, 0.1) * EnvGen.kr(Env.perc(0.01, 0.1), tremoloTrig);
129    //feedback
130    feedback = CombC.ar(sig, 0.2, tremRate.reciprocal, 5);
131    fade = feedback * EnvGen.kr(Env.asr(15, 1, 15, \sine), gate) * amp * 0.75;
132    Out.ar("interludeTremoloBus", fade);
133  }).add;
134
135  //-----gen music
136  genPatterns = {arg guitarSeqIn, accompLowSeqIn, accompHighSeqIn, sectionSeqIn, beatFrac = 1/8;
137    var calcSustains, genSectionSec, sectionLimits, measureCount;
138
139    //-----helper sus function
140    calcSustains = {arg stringSeq, durSeq;
141      var res = [];
142      stringSeq.size.do({arg index;
143        var curString, dur, count;
144        if(stringSeq[index].isRest.not, {
145          curString = stringSeq[index];
146          dur = durSeq[index];
147          count = 1;
148          while({(stringSeq[(index + count).clip(0, stringSeq.size - 1)] != curString) &&
149            (dur < 16) && (count < 100)}, {
150            dur = dur + durSeq[(index + count).clip(0, durSeq.size - 1)];
151            count = count + 1;
152          });
153          res = res.add(dur.clip(0, 16));
154        }, {
155          res.add(Rest());
156        });
157      };
158      res
159    };
160
161    genSectionSec = {arg seq, startTime, endTime, type;
162      var durSum, resSeqs, inSecs, mult;
163      durSum = 0;
164      resSeqs = [];
165      seq.do({arg item;
166        if((durSum >= startTime) && (durSum < endTime), {
167          var dur = durSum - startTime;
168          if((resSeqs.size == 0) && (dur > 0), {
169            switch(type,
170              0, {resSeqs = resSeqs.add([Rest(-1), Rest(-1), dur])},
171              1, {resSeqs = resSeqs.add([Rest(-1), Rest(-1), dur])},
172              2, {resSeqs = resSeqs.add([Rest(-1), dur, dur])}
173            );
174            resSeqs = resSeqs.add(item);
175          });
176          durSum = durSum + if(type == 2, {item[1]}, {item[2]});
177        });
178      };
179      resSeqs
180    };
181
182    measureCount = 0;
183    sectionLimits = [];
184    sectionSeqIn.slice(nil, 0).add(100000).doAdjacentPairs({arg a, b; sectionLimits = sectionLimits.add([a, b])});
185    `sectionStartMeasure = [];
186    sectionLimits.collect({arg timePair, secIndex;
187      var startTime, endTime, beatLength, beatSeq, measureSeq,
188      guitarSecSeq, accompLowSecSeq, accompHighSecSeq,
189      stringSeq, fretSeq, harmLimit, freqSeq, durSeq, susSeq, trigSeq, openStrings, pattern;
190
191      startTime = timePair[0];
192      endTime = timePair[1];
193
194      if((secIndex % 4) == 0, {measureCount = 0});
195      beatLength = (endTime - startTime) / 8;
196      beatSeq = ((beatLength / 2) - 1).asInteger.collect({[1, 2]});
197      beatSeq = if((beatLength % 2) == 0, {beatSeq.add([1, 2])}, {beatSeq.add([1, 2, 3])});
198      measureSeq = measureCount + beatSeq.collect({arg measure, mIndex; measure.collect({mIndex + 1})}).flat;
199      `sectionStartMeasure = `sectionStartMeasure.add(measureCount + 1);
200      measureCount = measureSeq.last;
201      beatSeq = beatSeq.flat;
202      measureSeq = measureSeq.add(0);
203      beatSeq = beatSeq.add(0);

```

```

204 guitarSecSeq = genSectionSec.value(guitarSeqIn, startTime, endTime, 0);
205 accompLowSecSeq = genSectionSec.value(accompLowSeqIn, startTime, endTime, 1);
206 accompHighSecSeq = genSectionSec.value(accompHighSeqIn, startTime, endTime, 2);
207
208 if(accompHighSecSeq == [], {accompHighSecSeq = [[Rest(-1), 1, 0], [Rest(-1), 1, 0]]});
209
210 openStrings = [1/1, 3/2, 2/1, 5/2, 35/12, 7/2];
211 harmLimit = [9, 8, 7, 6, 5, 4];
212 stringSeq = guitarSecSeq.slice(nil, 0);
213 fretSeq = guitarSecSeq.slice(nil, 1);
214 durSeq = guitarSecSeq.slice(nil, 2);
215 susSeq = calcSustains.value(stringSeq, durSeq);
216 freqSeq = stringSeq.collect({arg string, index;
217   if(string.isRest, {Rest(0)}, {
218     var midi, freq;
219     //this is transposed up because karplus-strong does not really sound correctly in the guitar range
220     midi = (62.midicps * openStrings[string]).cpsmidi + fretSeq[index];
221     freq = midi.midicps * if((secIndex % 4) != 3, {1}, {[1, harmLimit[string].rand + 1].choose}));
222   });
223
224 pattern = EventPatternProxy.new;
225 pattern.source = Ppar([
226   Pbind(
227     \instrument, \karplus ++ `hash,
228     \amp, 0.3,
229     \dur, Pseq(durSeq * beatFrac),
230     \sustain, Pseq(susSeq * beatFrac),
231     \freq, Pseq(freqSeq),
232     \bus, `guitarBus.index),
233   if(accompLowSecSeq.size > 1, {
234     Pmono(
235       \accompBass ++ `hash,
236       \amp, 0.5,
237       \freq1, Pseq(accompLowSecSeq.slice(nil, 0)),
238       \freq2, Pseq(accompLowSecSeq.slice(nil, 1)),
239       \dur, Pseq(accompLowSecSeq.slice(nil, 2) * beatFrac),
240       \busLower, if(secIndex % 2 == 0, {`accompLowLowerBusA.index}, {`accompLowLowerBusB.index}),
241       \busUpper, if(secIndex % 2 == 0, {`accompLowUpperBusA.index}, {`accompLowUpperBusB.index}))
242   }, {
243     Pmono(
244       \accompBass ++ `hash,
245       \amp, 0.5,
246       \freq1, Pseq([accompLowSecSeq[0][0]]),
247       \freq2, Pseq([accompLowSecSeq[0][1]]),
248       \dur, Pseq([accompLowSecSeq[0][2]] * beatFrac),
249       \busLower, if(secIndex % 2 == 0, {`accompLowLowerBusA.index}, {`accompLowLowerBusB.index}),
250       \busUpper, if(secIndex % 2 == 0, {`accompLowUpperBusA.index}, {`accompLowUpperBusB.index}))
251   }),
252   Pbind(
253     \instrument, \accompTreble ++ `hash,
254     //\freq, Pseq(accompHighSecSeq.slice(nil, 0)),
255     \freq, Pseq(accompHighSecSeq.slice(nil, 0).curdle(0.3).collect({arg item; item.cpsmidi - 0.16 + 0.32.rand}).midicps.flat),
256     \dur, Pseq(accompHighSecSeq.slice(nil, 1) * beatFrac),
257     \sustain, Pseq(accompHighSecSeq.slice(nil, 2) * beatFrac),
258     \amp, 0.5,
259     \bus, `accompHighBus.index),
260   Pbind(
261     \instrument, \transport ++ `hash,
262     \measure, Pseq(measureSeq),
263     \beat, Pseq(beatSeq),
264     \dur, beatFrac * 8
265   ),
266   Pbind(
267     \instrument, \click ++ `hash,
268     \beat, Pseq(beatSeq.drop(-1)),
269     \dur, beatFrac * 8
270   )
271 );
272 pattern
273 ];
274 };
275
276 )

```

cicc_transcriber.scd

```

1 (
2 `transcribe = {arg scoreData, seed;
3   var rawMusicData, timeSigData, sectionData, dir, basePath, scoreFile, maxSize, lineBreakString, openStrings, musicData;
4
5   rawMusicData = scoreData[0];
6   timeSigData = scoreData[1];
7   sectionData = scoreData[2];
8
9   basePath = `dir ++/ ".." ++/ "lilypond" ++/ "seed." ++ seed;
10  basePath.mkdir;
11  (basePath ++/ "includes").mkdir;
12
13  scoreFile = File(basePath ++/ "cicc_score.ly".standardizePath, "w");
14  scoreFile.write(File.readAllString(basePath ++/ "templates" ++/ "cicc_score_template.ly").replace("seed: xxx", "seed: " ++ seed));
15  scoreFile.close;
16  scoreFile = File(basePath ++/ "cicc_pseudoincidents_def.ly".standardizePath, "w");
17  scoreFile.write(File.readAllString(basePath ++/ "templates" ++/ "cicc_pseudoincidents_def.ly"));
18  scoreFile.close;
19
20  openStrings = [1/1, 3/2, 2/1, 5/2, 35/12, 7/2];
21
22  maxSize = 0;
23  musicData = rawMusicData.collect({arg partData, p;
24    var res;
25    res = partData.collect({arg item, i;
26      var note, rest;
27      switch(p,
28        0, {
29          var string, fret, dur, sus;
30          string = item[0];
31          fret = item[1];
32          dur = item[2];
33          sus = item[3];
34          note = sus.collect({[string, fret, i]});
35        },
36        1, {
37          var freq, dur, sus;
38          freq = item[0];
39          dur = item[1];
40          sus = item[2];
41          note = sus.collect({[freq, i]});
42          rest = if(p < rawMusicData.size, {(dur - sus).collect({[-1, i]}), {[]}};
43        },
44        2, {
45          var freq1, freq2, dur, sus;
46          freq1 = item[0];
47          freq2 = item[1];

```

```

48         dur = item[2];
49         sus = 4;
50         note = sus.collect({[freq1, freq2 - freq1], i});
51         rest = if(p < rawMusicData.size, {(dur - sus).collect({[-1, i])}, {});
52     }
53 };
54     note ++ rest
55 }).flatten;
56     if(res.size > maxSize, {maxSize = res.size});
57     res
58 });
59
60 musicData = musicData.collect({arg partData, p;
61     var lastSectionSize, lastSectionSizeTrunc, finalSectionSize, ext;
62     lastSectionSize = (maxSize - sectionData.last[0]);
63     lastSectionSizeTrunc = lastSectionSize.trunc(16);
64     finalSectionSize = if(lastSectionSize != lastSectionSizeTrunc, {lastSectionSizeTrunc + 16}, {lastSectionSize});
65     ext = finalSectionSize - lastSectionSize;
66     partData.extend((maxSize + ext), if(p == 0, {partData.last}, {[-1, partData.last[1]]}));
67 });
68
69 lineBreakString = "";
70 sectionData.slice(nil, 0).add(musicData[0].size).differentiate.drop(1).clump(4).do({arg section;
71     var remainder, endSec;
72     remainder = 0;
73
74     section.do({arg len, index;
75         var noFullSystems;
76
77         //this causes a problem if a section is less than 10 half notes (4 measures)
78         if(remainder % 16 == 0, {
79             lineBreakString = lineBreakString ++ "\\repeat unfold 8 {s2 \\noBreak} \\break \\n";
80         }, {
81             var noBeats;
82             noBeats = ((remainder + (64 - remainder).trunc(16)) / 8).asInteger;
83             lineBreakString = lineBreakString ++ "\\pseudoIndents 0 " ++
84                 (21 * (8 - noBeats)) ++ " \\repeat unfold " ++ noBeats ++ " {s2 \\noBreak} \\break \\n";
85         });
86
87         remainder = len - (64 - remainder).trunc(16);
88
89         noFullSystems = (remainder.trunc(64) / 64).asInteger;
90         if(noFullSystems > 0, {
91             (noFullSystems - 1).do({
92                 lineBreakString = lineBreakString ++ "\\repeat unfold 8 {s2 \\noBreak} \\break \\n";
93             });
94             if(remainder % 64 != 8, {
95                 lineBreakString = lineBreakString ++ "\\repeat unfold 8 {s2 \\noBreak} \\break \\n";
96                 remainder = remainder - (noFullSystems * 64);
97             }, {
98                 lineBreakString = lineBreakString ++ "\\pseudoIndents 0 42 \\repeat unfold 6 {s2 \\noBreak} \\break \\n";
99                 remainder = 24;
100             });
101         });
102     });
103
104     if(remainder > 0, {
105         lineBreakString = lineBreakString ++ "\\pseudoIndents 0 " ++
106             (21 * (8 - (remainder / 8).asInteger)) ++ " \\repeat unfold " ++ (remainder / 8).asInteger ++ " {s2 \\noBreak} \\break \\n";
107     });
108 });
109
110 musicData.do({arg part, p;
111     var amps, harm, modi, timeSigIndex, sectionCount, sectionIndex, subSectionIndex, curTimeSig,
112     lilyFile, lilyString, voices, lastVal, lilyNotes, lilyOcts, lilyGString, isHarmonic, measureCount,
113     lilyNote, lilyDur, lilyRest, lilyBeatingMark, curTime = 0, noteTuples, markupSuffixes;
114
115     //create file
116     lilyFile = switch(p,
117         0, {File(basePath ++ "includes" ++ "cicc.guitar.ly".standardizePath, "w")},
118         1, {File(basePath ++ "includes" ++ "cicc.high.ly".standardizePath, "w")},
119         2, {File(basePath ++ "includes" ++ "cicc.low.ly".standardizePath, "w")});
120
121 //start lilypond directives
122 lilyString = "";
123
124 lastVal = nil;
125
126 //start voice
127 lilyString = lilyString ++ "\n{ ";
128 lilyString = lilyString ++ "\n\\set Score.markFormatter = #format-mark-box-numbers ";
129
130 lilyString = lilyString ++ "\\tempo \\markup {\\concat {\\smaller \\general-align #Y #DOWN \\note #\"2\" #1 \\normal-text \" = approx. 90\"}}";
131 if(p != 1, {lilyString = lilyString ++ "\\override Staff.TimeSignature #'stencil = ##f});
132 lilyString = lilyString ++ "\\numericTimeSignature \\time 2/2\\n";
133
134 lilyString = switch(p,
135     0, {lilyString ++ "\\clef \"treble.(8)\\n\"},
136     1, {lilyString ++ "\\clef \"treble\"(8)\\n\"},
137     2, {lilyString ++ "\\clef \"bass.(8)\\n\"});
138
139 lilyNotes = ["c", "cis", "d", "dis", "e", "f", "fis", "g", "gis", "a", "ais", "b"];
140 lilyOcts = ["", "1", "2", "3", "4", "5", "6", "7", "8"];
141
142 timeSigIndex = 0;
143 sectionCount = 0;
144 sectionIndex = 1;
145 subSectionIndex = 1;
146 curTimeSig = 4;
147 measureCount = 0;
148 part.clump(4).do({arg beat, i;
149     var gSum = 0;
150
151     beat.separate({arg a, b; ((a[0] != -1) || (b[0] != -1)) && (a != b)}.do({arg group, g; var noteLength, target = 0;
152         noteLength = group.size;
153         gSum = gSum + noteLength;
154
155         //add ties
156         lilyString = lilyString ++ if((p != 2) && (group[0] == lastVal) && (group[0][0] != -1), {"- "}, {" "});
157         //add barcheck count
158         lilyString = lilyString ++ if((curTime % curTimeSig == 0) && (i != 0), {measureCount = measureCount + 1; " | "}, {" "});
159
160         if((i == (sectionData[sectionCount][0] / 4)) && (g == 0), {
161             var barType, pageBreak;
162             barType = switch(sectionData[sectionCount][1],
163                 0, {"\\|\\n"},
164                 1, {"\\|\\n"},
165                 -1, {"\\|\\n" ++ "\\set Score.currentBarNumber = #1 "});
166             pageBreak = switch(sectionData[sectionCount][1], 0, {" "}, 1, {" "}, -1, {measureCount = 0; "\\n\\pageBreak \\n \\time 2/2\\n"});
167             isHarmonic = switch(sectionData[sectionCount][1], 0, {false}, 1, {true}, -1, {false});
168             lilyString = lilyString ++ "\\bar " ++ barType ++
169                 " \\mark \\markup { \\bold \\box " ++ sectionIndex ++ " " ++ subSectionIndex ++ " } " ++ pageBreak;

```

```

173         if(sectionCount < (sectionData.size - 1), {sectionCount = sectionCount + 1});
174         switch(sectionData[sectionCount][1],
175             0, {subSectionIndex = subSectionIndex + 1},
176             1, {subSectionIndex = subSectionIndex + 1},
177             -1, {sectionIndex = sectionIndex + 1; subSectionIndex = 1})
178     });
179
180     if((i == (timeSigData[timeSigIndex][0] / 4)) && (g == 0), {
181         timeSigData[timeSigIndex][0];
182         curTimeSig = timeSigData[timeSigIndex][1];
183         if(curTimeSig % 2 == 0, {
184             lilyString = lilyString + "\n\\time " + (curTimeSig / 2).asInteger.asString + "/2\n";
185         }, {
186             lilyString = lilyString + "\n\\time " + curTimeSig.asInteger.asString + "/4\n";
187         });
188         if(timeSigIndex < (timeSigData.size - 1), {timeSigIndex = timeSigIndex + 1});
189         curTime = 0;
190     });
191
192     switch(p,
193         0, {
194             lilyNote = lilyNotes[(((38.midicps * openStrings[group[0][0]]).cpsmidi + group[0][1]).round(1) % 12)];
195             lilyNote = lilyNote + lilyOcts[(((38.midicps * openStrings[group[0][0]]).cpsmidi + group[0][1]).round(1) / 12).asInteger - 2];
196         },
197         1, {
198             if(group[0][0] != -1, {
199                 lilyNote = lilyNotes[(((group[0][0].cpsmidi).round(1) % 12)];
200                 lilyNote = lilyNote + lilyOcts[(((group[0][0]).cpsmidi).round(1) / 12).asInteger - 2];
201             }, {lilyNote = "r"});
202         },
203         2, {
204             if(group[0][0] != -1, {
205                 lilyNote = lilyNotes[(((group[0][0][0].cpsmidi).round(1) % 12)]; // * 2;
206                 lilyNote = lilyNote + lilyOcts[(((group[0][0][0]).cpsmidi).round(1) / 12).asInteger - 2];
207                 lilyBeatingMark = "" \\markup{ " ++ group[0][0][1].round(0.1) ++ " } ";
208             }, {lilyNote = "r"});
209         }
210     );
211
212     //duration
213     lilyDur = switch(noteLength, 1, {"16 "}, 2, {"8 "}, 3, {"8. "}, 4, {"4 "});
214     //append rest directive
215     //lilyRest = "";
216     lilyGString = if(((group[0] != lastVal) && (p == 0)), {
217         var stringString, fretString;
218         stringString = ["VI ", "V ", "IV ", "III ", "II ", "I "][group[0][0]];
219         fretString = group[0][1].asString;
220         if(!isHarmonic, {fretString = "\\musicglyph \"noteheads.s0harmonic\""});
221         "\\markup{\\concat{ " ++ stringString ++ " \\super " ++ fretString ++ " } } "
222     }, {"r"});
223
224     if((p != 2) || (lilyNote == "r"), {
225         lilyString = lilyString + lilyNote + lilyDur + lilyGString;
226     }, {
227         lilyString = lilyString + "<<{ " ++ lilyNote + lilyDur ++
228         " \\laissezVibrer " ++ lilyBeatingMark ++ " }\\ \\new Voice { \\voiceTwo " ++
229         lilyNote ++ lilyDur ++ " \\laissezVibrer }>> \\oneVoice " ++ lilyGString;
230     });
231
232     //beam group
233     if((p != 2) && (g == 0) && (noteLength != 4), {lilyString = lilyString + " [ "});
234     if((p != 2) && (gSum == 4) && (noteLength != 4), {lilyString = lilyString + " ] "});
235
236     lastVal = group[0];
237     curTime = curTime + (noteLength / 4);
238
239 });
240
241 //end voice
242 lilyString = lilyString + " ] \\bar \".\\n" } \\n";
243
244
245 noteTuples = [lilyNotes, lilyOcts].allTuples.collect({arg val; val.join}).join("|");
246
247 markupSuffixes = ["VI ", "V ", "IV ", "III ", "II ", "I "].collect({arg stringString;
248     ("\\musicglyph \\noteheads.s0harmonic\\\"\" ++ (0..14)).collect({arg fret;
249     "\\markup{\\concat{ " ++ stringString ++ " \\super " ++ fret.asString ++ " } }")}).flatten.join("|");
250
251 lilyString.findRegexp(
252     "(" ++ noteTuples ++ ")4 (" ++ markupSuffixes ++ ") ^ " ++
253     "(" ++ noteTuples ++ ")4 ^ (" ++ noteTuples ++ ")4 ^ (" ++
254     noteTuples ++ ")4 ^ (" ++ noteTuples ++ ")4 ^ (" ++ noteTuples ++ ")4"
255 ).clump(8).do({arg match;
256     lilyString = lilyString.replace(match[0][1], match[1][1] ++ "1. " ++ match[2][1]);});
257
258 lilyString.findRegexp(
259     "(" ++ noteTuples ++ ")4 (" ++ markupSuffixes ++ ") ^ (" ++ noteTuples ++ ")4 ^ (" ++ noteTuples ++ ")4 ^ (" ++ noteTuples ++ ")4"
260 ).clump(6).do({arg match;
261     lilyString = lilyString.replace(match[0][1], match[1][1] ++ "1 " ++ match[2][1]);});
262
263 lilyString.findRegexp(
264     "(" ++ noteTuples ++ ")4 (" ++ markupSuffixes ++ ") ^ (" ++ noteTuples ++ ")4 ^ (" ++ noteTuples ++ ")4"
265 ).clump(5).do({arg match;
266     lilyString = lilyString.replace(match[0][1], match[1][1] ++ "2. " ++ match[2][1]);});
267
268 lilyString.findRegexp("(" ++ noteTuples ++ ")4 (" ++ markupSuffixes ++ ") ^ (" ++ noteTuples ++ ")4").clump(4).do({arg match;
269     lilyString = lilyString.replace(match[0][1], match[1][1] ++ "2 " ++ match[2][1]);});
270
271
272 //consolidate notes
273 lilyString.findRegexp(
274     "(" ++ noteTuples ++ ")4 ^ (" ++ noteTuples ++ ")4 ^ (" ++
275     noteTuples ++ ")4 ^ (" ++ noteTuples ++ ")4 ^ (" ++ noteTuples ++ ")4 ^ (" ++ noteTuples ++ ")4"
276 ).clump(7).do({arg match;
277     lilyString = lilyString.replace(match[0][1], match[1][1] ++ "1.");});
278
279 lilyString.findRegexp(
280     "(" ++ noteTuples ++ ")4 ^ (" ++ noteTuples ++ ")4 ^ (" ++ noteTuples ++ ")4 ^ (" ++ noteTuples ++ ")4"
281 ).clump(5).do({arg match;
282     lilyString = lilyString.replace(match[0][1], match[1][1] ++ "1");});
283
284 lilyString.findRegexp("(" ++ noteTuples ++ ")4 ^ (" ++ noteTuples ++ ")4 ^ (" ++ noteTuples ++ ")4").clump(4).do({arg match;
285     lilyString = lilyString.replace(match[0][1], match[1][1] ++ "2.");});
286
287 lilyString.findRegexp("(" ++ noteTuples ++ ")4 ^ (" ++ noteTuples ++ ")4").clump(3).do({arg match;
288     lilyString = lilyString.replace(match[0][1], match[1][1] ++ "2");});
289
290
291 //consolidate rests
292 lilyString.findRegexp("r4 r4 r4 r4").clump(2).do({arg match;
293     lilyString = lilyString.replace(match[0][1], "R1*3/2");});
294
295 lilyString.findRegexp("r4 r4 r4 r4").clump(2).do({arg match;
296     lilyString = lilyString.replace(match[0][1], "r4 r1");});
297

```

```

298 lilyString.findRegexp("r4 r4 r4").clump(2).do({arg match;
299   lilyString = lilyString.replace(match[0][1], "r1");});
300
301 lilyString.findRegexp("r4 r4 r4").clump(2).do({arg match;
302   lilyString = lilyString.replace(match[0][1], "r2.");});
303
304 lilyString.findRegexp("r4 r4").clump(2).do({arg match;
305   lilyString = lilyString.replace(match[0][1], "r2.");});
306
307 lilyString.findRegexp("\\| r1").clump(2).do({arg match;
308   lilyString = lilyString.replace(match[0][1], "| R1");});
309
310 lilyString.findRegexp("d\\nrl").clump(2).do({arg match;
311   lilyString = lilyString.replace(match[0][1], "2\\n R1");});
312
313 //write file
314 lilyFile.write("{\\n" ++ lineBreakString ++ "}\\n" ++ lilyString);
315 lilyFile.close;
316 });
317 };
318
319
320 `genScoreData = {arg guitarSeq, accompLowSeq, accompHighSeq, timeSigInsSeq, sectionSeq;
321   var stringSeq, fretSeq, durSeq,
322     partData, timeSigData, sectionData;
323   stringSeq = guitarSeq.slice(nil, 0);
324   fretSeq = guitarSeq.slice(nil, 1);
325   durSeq = guitarSeq.slice(nil, 2);
326   partData = [
327     [stringSeq, fretSeq, durSeq, durSeq].flop,
328     accompHighSeq,
329     accompLowSeq
330   ];
331   timeSigData = timeSigInsSeq;
332   sectionData = sectionSeq;
333   [partData, timeSigData, sectionData]
334 };
335 )

```

cicc_gui.scd

```

1 (
2   //~FUNCTION THAT GENERATES THE GUI
3   `generateGUI = {
4     var win, clockStringFunc, metronomeStringFunc, metronomeColorFunc, masterView, faderView, helpView, tabs;
5     var tabButtonReset, transportButton, mixerButton, helpButton, startPos = 0;
6     var partAbbr = ["guitar", "accompHigh", "accompLowLower", "accompLowUpper", "interlude", "click"];
7     var trackNames = ["guitar", "high", "low 1", "low 2", "interlude", "click"];
8     var partVols, partMutes, partPans;
9     var masterMute, masterVol;
10
11     // set initial mixer values
12     partVols = [1, 1, 1, 1, 1, 1];
13     partMutes = [0, 1, 1, 1, 1, 0];
14     partPans = [0, 0, 0, 0, 0, 0];
15     masterMute = 1;
16     masterVol = 1;
17
18     // these funcs update the elements of the transport panel
19     clockStringFunc = {
20       arg measure, beat;
21       var measureString, beatString, leadSpace;
22       measureString = measure.asInteger.asString;
23       beatString = beat.asInteger.asString;
24       leadSpace = (3 - measureString.size).collect({" "}).join;
25       leadSpace ++ measureString ++ "." ++ beatString
26     };
27     // [-30, -105, -104].asAscii and [-30, -105, -113].asAscii are unicode inverse bullet and normal bullet, respectively
28     metronomeStringFunc = { arg beat; if(beat == 1, {[-30, -105, -104].asAscii}, {[-30, -105, -113].asAscii}) };
29     metronomeColorFunc = { arg beat; if(beat == 1, {Color.red}, {Color.black}) };
30
31     win = Window("Counterfeiting in Colonial Connecticut", Rect(500, 500, 1100, 575), false).front;
32     masterView = {
33       var updateTransport, updateSection,
34       view, generator, transport, countOff, ranSeed, order, tempo, sectionDisplay, clock, metronome, address;
35
36       // this func updates the whole transport panel
37       updateTransport = {arg measure, beat;
38         clock.string = clockStringFunc.value(measure, beat);
39         metronome.stringColor = metronomeColorFunc.value(beat);
40         metronome.string = metronomeStringFunc.value(beat);
41         {0.75.wait; {metronome.string = ""}.defer}.fork(`tempoClock, quant: 0);
42       }.inEnvir;
43
44       // this func handles the movement between sections
45       updateSection = {arg shift, stop = true, manualCall = true;
46         var runThis;
47         runThis = (manualCall || (manualCall.not && `autoAdvance));
48         runThis = runThis && ((`currentSection + shift) < `sectionOrder.size);
49         runThis = runThis && ((`currentSection % 4) == 3) && `interludes && manualCall.not;
50         if(runThis, {
51           var truncOnly, section, subSection;
52           if(`isPlaying, {
53             if(stop, {
54               `patterns[`sectionOrder[`currentSection]].stop
55             })
56           })
57
58           truncOnly = case
59             {(`currentSection + shift) < 0} {true}
60             {(shift < 0) && `isPlaying} {true}
61             {(shift < -1) && ((`currentSection % 4) > 0)} {true}
62             {true} {false};
63
64           if(truncOnly.not, {
65             `currentSection = (`currentSection + shift).trunc(shift.abs);
66           }, {
67             `currentSection = `currentSection.trunc(shift.abs);
68           });
69
70           section = ((`sectionOrder[`currentSection] / 4) + 1).asInteger;
71           subSection = ((`sectionOrder[`currentSection] % 4) + 1).asInteger;
72           sectionDisplay.string = "section: " ++ section.asString ++ "." ++ subSection.asString;
73           if(`isPlaying, {
74             countOff = {
75               if(`interludes && ((`currentSection % 4) == 0), {
76                 Pbind(
77                   \instrument, \click ++ `hash,
78                   \beat, Pseq([1, 2, 1, 2]),
79                   \dur, 1
80                 ).play(`tempoClock, quant: 0);
81                 [1, 2, 1, 2].do({arg beat;
82                   {

```

```

83         metronome.stringColor = metronomeColorFunc.value(beat);
84         metronome.string = metronomeStringFunc.value(beat);
85     }.defer;
86     0.75.wait;
87     {metronome.string = ""}.defer;
88     0.25.wait;
89     });
90
91     play.set(\sel, ~currentSection % 2);
92     patterns["sectionOrder["~currentSection]].play(~tempoClock, quant: 0);
93     if(~interludes && ((~currentSection % 4) == 3) && (~currentSection != (~sectionOrder.size - 1)), {
94         var center, interval, freq1, freq2, tremRate;
95         center = 50 - 12.0.rand;
96         interval = 3.0.rand + 2;
97         freq1 = (center + (interval / 2)).midicps;
98         freq2 = (center - (interval / 2)).midicps;
99         tremRate = 50 + 4.0.rand2;
100         ~interludeTremelo.set(\gate, 1, \amp, 1, \freq1, freq1, \freq2, freq2, \tremRate, tremRate);
101     });
102     if((~currentSection % 4) == 0, {
103         ~interludeTremelo.set(\gate, 0);
104     });
105     if((~currentSection % 4) != 0 && (~currentSection % 4) != 3), {
106         ~interludeTremelo.set(\gate, 0, \amp, 0);
107     });
108     }.fork(~tempoClock, quant: 0);
109
110     }, {
111         var measure, beat;
112         measure = ~sectionStartMeasure["sectionOrder["~currentSection]];
113         beat = 1;
114         updateTransport.value(measure, beat);
115     });
116
117     }.inEnvir;
118
119     // these funcs receive messages from the synth
120     OSCFunc({ arg msg, time;
121     {
122         var measure, beat;
123         measure = msg[3];
124         beat = msg[4];
125         updateTransport.value(measure, beat)
126     }.inEnvir.defer;
127     }, '~measureClock' ++ ~hash, s.addr);
128
129     OSCFunc({ arg msg, time; {updateSection.value(1, false, false)}.inEnvir.defer}, '~nextSubsection' ++ ~hash, s.addr);
130
131     OSCdef(\externalAdvance ++ ~hash, {arg msg, time; {updateSection.value(1)}.inEnvir.defer}, '~nextSubsection', s.addr);
132
133     view = View(win);
134     generator = HLayout (
135         ranSeed = TextField(view, Rect(10, 10, 10, 20)).string("20200525"),
136         Button(view).states(["reset seed"]).action.({ ranSeed.string = "20200525".inEnvir),
137         Button(view).states(["random seed"]).action.({ ranSeed.string = 50000000.rand.asString.inEnvir),
138         Button(view).states(["generate"]).action.({
139             {genAll.value(ranSeed.string.asInteger); ~appStatus.string = "status: ready".fork(AppClock);
140             ~appStatus.string = "status: generating".inEnvir),
141         Button(view).states(["transcribe"]).action.({
142             {transcribe.value(scoreData, ranSeed.string); ~appStatus.string = "status: ready".fork(AppClock);
143             ~appStatus.string = "status: transcribing".inEnvir),
144         ~appStatus = StaticText(view).string("status: ready"), stretch: 1, nil);
145     transport = HLayout (
146         Button(view).states(["<", Color.black]).action.({arg pState; updateSection.value(-4)}.inEnvir),
147         Button(view).states(["<", Color.black]).action.({arg pState; updateSection.value(-1)}.inEnvir),
148         Button(view).states(["play", Color.black], ["stop", Color.black, Color.grey]).action.({
149             {arg pState;
150             if(pState.value == 0, {
151                 var measure, beat;
152                 countOff.stop;
153                 ~isPlaying = false;
154                 patterns["sectionOrder["~currentSection]].stop;
155                 ~interludeTremelo.set(\gate, 0);
156                 measure = ~sectionStartMeasure["~currentSection"];
157                 beat = 1;
158                 updateTransport.value(measure, beat);
159                 ~interludeTremelo.set(\gate, 0, \amp, 0);
160             }},
161             countOff = {
162                 Pbind(
163                     \instrument, \click ++ ~hash,
164                     \beat, Pseq([1, 2, 1, 2]),
165                     \dur, 1
166                 ).play(~tempoClock, quant: 0);
167                 [1, 2, 1, 2].do({arg beat;
168                     {
169                         metronome.stringColor = metronomeColorFunc.value(beat);
170                         metronome.string = metronomeStringFunc.value(beat);
171                     }.defer;
172                     0.75.wait;
173                     {metronome.string = ""}.defer;
174                     0.25.wait;
175                 });
176                 ~isPlaying = true;
177                 play.set(\sel, ~currentSection % 2);
178                 patterns["sectionOrder["~currentSection]].play(~tempoClock, quant: 0);
179                 if(~interludes && ((~currentSection % 4) == 3) && (~currentSection != (~sectionOrder.size - 1)), {
180                     var center, interval, freq1, freq2, tremRate;
181                     center = 50 - 12.0.rand;
182                     interval = 3.0.rand + 2;
183                     freq1 = (center + (interval / 2)).midicps;
184                     freq2 = (center - (interval / 2)).midicps;
185                     tremRate = 50 + 4.0.rand2;
186                     ~interludeTremelo.set(\gate, 1, \amp, 1, \freq1, freq1, \freq2, freq2, \tremRate, tremRate);
187                 });
188                 }.fork(~tempoClock, quant: 0);
189             })
190         }.inEnvir
191     ),
192     Button(view).states([">", Color.black]).action.({arg pState; updateSection.value(1)}.inEnvir),
193     Button(view).states([">>", Color.black]).action.({arg pState; updateSection.value(4)}.inEnvir), nil,
194     sectionDisplay = StaticText(win).string("section: 1.1").font(Font("Monaco", 70)), nil);
195     view.layout(HLayout (
196         VLayout (
197             HLayout (clock = StaticText(win).string(" 1.1").font(Font("Monaco", 200)),
198                 StaticText(win).string("[")font(Font("Monaco", 200)),
199                 metronome = StaticText(win).string([-30, -105, -104].asAscii).font(Font("Monaco", 300)).stringColor(Color.red)),
200                 nil, transport, nil,
201             HLayout (
202                 tempo = TextField(view).string("90"),
203                 Button(view).states(["set tempo"]).action.({~tempoClock.tempo = tempo.string.asInteger / 60}.inEnvir),
204                 StaticText(view).string(" | "),
205                 Button(view).states(["auto advance", Color.black], ["auto advance", Color.black, Color.grey]).action.({
206                     arg v; ~autoAdvance = if(v.value == 0, {false}, {true}); ~autoAdvance;
207                 }.inEnvir).value(1),
208                 Button(view).states(["interludes", Color.black], ["interludes", Color.black, Color.grey]).action.({

```

```

208     arg v; interludes = if(v.value == 0, {false}, {true})
209   }.inEnvir),
210   [StaticText(view).string(" | "),
211   address = TextField(view, Rect(10, 10, 10, 20)).string("127.0.0.1:57120"),
212   Button(view).states(["set address:port"]).action({
213     var addr, ip, port;
214     addr = address.string.split($:);
215     ip = addr[0];
216     port = addr[1].asInteger;
217     thisProcess.openUDPPort(port);
218     addr = NetAddr(ip, port);
219     OSCdef(\externalAdvance ++ ~hash, {arg msg, time; {updateSection.value(1)}.inEnvir.defer}, '/nextSubsection', addr);
220   }.inEnvir),
221   [StaticText(view).string(" "), stretch: 1]),
222   [StaticText(view).string(" "), stretch: 1],
223   HLayout (
224     order = TextField(view).string("1-16"),
225     Button(view).states(["set order"]).action({
226       patterns["sectionOrder["currentSection]].stop;
227       ~sectionOrder = order.string.split($,).collect({arg secEntry;
228         var bounds;
229         bounds = secEntry.split($-).collect({arg item; item.asInteger - 1});
230         ((bounds.minItem)..(bounds.maxItem)).collect({arg sec;
231           (sec.asInteger * 4) + [0, 1, 2, 3]
232         })});
233     }).flat;
234     ~currentSection = 0;
235     updateSection.value(0);
236   }.inEnvir),
237   [StaticText(view).string(" "), stretch: 1]),
238   [StaticText(view).string(" "), stretch: 1], generator
239 ), alignment: \top));
240 faderView = {
241   var view, masterIndicators, trackIndicators, master, tracks;
242   view = View(win);
243   masterIndicators = {LevelIndicator()} ! 2;
244   trackIndicators = {LevelIndicator()} ! 6;
245
246   OSCFunc.new({arg msg; {
247     {arg i; masterIndicators[i].value = msg[3 + i].ampdb.linlin(-40, 0, 0, 1)} ! 2}.defer},
248   '/masterLevels' ++ ~hash, s.addr);
249   OSCFunc.new({arg msg; {
250     {arg i; trackIndicators[i].value = msg[3 + i].ampdb.linlin(-40, 0, 0, 1)} ! 6}.defer},
251   '/trackLevels' ++ ~hash, s.addr);
252
253   master = HLayout (
254     VLayout (
255       HLayout (
256         Slider(view).value.(0.8).action({
257           {arg v; masterVol = v.value * 1.25; ~play.set(\masterVol, masterVol)}.inEnvir,
258           masterIndicators[0],
259           masterIndicators[1]), stretch: 2],
260         Button(view).states(["mute", Color.black], ["mute", Color.black, Color.grey]).action({
261           {arg v; masterMute = (1 - v.value).abs; ~play.set(\masterMute, masterMute)}.inEnvir,
262           StaticText(view).string("master")
263         ), nil);
264       tracks = {arg part;
265         HLayout (
266           VLayout (
267             HLayout (
268               Slider(view).value.(0.8).action({
269                 {arg v; partVols[part] = v.value * 1.25; ~play.set(partAbbr[part] ++ "Vol", partVols[part])}.inEnvir,
270                 trackIndicators[part]),
271               Button(view).states(["mute", Color.black], ["mute", Color.black, Color.grey]).action({
272                 {arg v; partMutes[part] = (1 - v.value).abs; ~play.set(partAbbr[part] ++ "Mute", partMutes[part])}.inEnvir).value.(
273                 if(part == 0 || (part == 5), {1}, {0}).value),
274               StaticText(view).string("pan").align(\center),
275               Knob(view).value.(0.5).action({
276                 {arg v; partPans[part] = v.value * 2 - 1; ~play.set(partAbbr[part] ++ "Pan", partPans[part])}.inEnvir,
277               StaticText(view).string(trackNames[part]).align(\center)
278             ), nil)
279           } ! 6;
280           view.layout.(HLayout(master, nil, *tracks));
281         helpView = {
282           TextView(win).string(File.readAllString("dir" ++ "cicc.readme.scd")).editable.(false);
283         };
284         tabButtonReset = {transportButton.value = 1; mixerButton.value = 1; helpButton.value = 1};
285         win.layout = VLayout (
286           HLayout (
287             HLayout (
288               [
289                 transportButton = Button().states(["transport", Color.white, Color.grey], ["transport", Color.black]).action({
290                   {tabButtonReset.value; transportButton.value = 0; tabs.index = 0 }.inEnvir).value.(0), stretch: 1
291               ], [
292                 mixerButton = Button().states(["mixer", Color.white, Color.grey], ["mixer", Color.black]).action({
293                   {tabButtonReset.value; mixerButton.value = 0; tabs.index = 1 }.inEnvir).value.(1), stretch: 1
294               ]
295             ),
296             helpButton = Button().states(["help", Color.white, Color.grey], ["help", Color.black]).action({
297               {tabButtonReset.value; helpButton.value = 0; tabs.index = 2 }.inEnvir).value.(1)
298           ),
299           tabs = StackLayout(masterView.value, faderView.value, helpView.value));
300       };
301     }
302   )

```

cicc_score_template.ly

```

1 \version "2.19.83"
2
3 \include "cicc.pseudoindeents.def.ly"
4
5 #(define factor 2)
6
7 #(define (enlarged-extent-laissez-vibrer::print grob)
8   (let* ((stil (laissez-vibrer::print grob))
9         (stil-ext (ly:stencil-extent stil X))
10        (stil-length (interval-length stil-ext))
11        (new-stil-length (* stil-length factor))
12        (scale-factor (/ new-stil-length stil-length))
13        (new-stil (ly:stencil-scale stil scale-factor 1))
14        (new-stil-ext (ly:stencil-extent new-stil X))
15        (x-corr (- (car stil-ext) (car new-stil-ext))))
16   (ly:stencil-translate-axis
17     new-stil
18     x-corr
19     X))
20
21 #(assoc-set! (assoc-ref all-grob-descriptions 'LaissezVibrerTie)
22 'stencil enlarged-extent-laissez-vibrer::print)
23
24 \paper {
25   #(set-paper-size "a4" 'portrait)

```

```

26 top-margin = 1 \cm
27 bottom-margin = 1 \cm
28 left-margin = 2.25 \cm
29 ragged-bottom = ##t
30
31 top-system-spacing =
32 #'((basic-distance . 20 )
33 (minimum-distance . 20 )
34 (padding . 0 )
35 (stretchability . 0))
36
37 system-system-spacing =
38 #'((basic-distance . 25 )
39 (minimum-distance . 25 )
40 (padding . 0 )
41 (stretchability . 0))
42
43 last-bottom-spacing =
44 #'((basic-distance . 15 )
45 (minimum-distance . 15 )
46 (padding . 0 )
47 (stretchability . 0))
48
49 systems-per-page = 5
50 first-page-number = 6
51 print-first-page-number = ##t
52
53 print-page-number = ##t
54 oddHeaderMarkup = \markup { \fill-line { \line { \on-the-fly #not-first-page {\italic {Counterfeiting in Colonial Connecticut} (seed: xxx)}}}}
55 evenHeaderMarkup = \markup { \fill-line { \line { \on-the-fly #not-first-page {\italic {Counterfeiting in Colonial Connecticut} (seed: xxx)}}}}
56 oddFooterMarkup = \markup { \fill-line {
57   \concat {
58     " "
59     \fontsize #1.5
60     \on-the-fly #print-page-number-check-first
61     \fromproperty #'page:page-number-string
62     " "}}}
63 evenFooterMarkup = \markup { \fill-line {
64   \concat {
65     " "
66     \fontsize #1.5
67     \on-the-fly #print-page-number-check-first
68     \fromproperty #'page:page-number-string
69     " "}}}
70 }
71
72 \header {
73   title = \markup { \italic {Counterfeiting in Colonial Connecticut}}
74   composer = \markup { \right-column {"Michael Winter" "(cdmx and gatlinburg, tennessee; 2020)"} }
75   poet = "seed: xxx"
76   tagline = ""
77 }
78
79 #(set-global-staff-size 11)
80
81 \layout {
82   indent = 0.0 \cm
83   line-width = 17 \cm
84   ragged-last = ##f
85   ragged-right = ##f
86
87   \context {
88     \Score
89     \override BarNumber.stencil = #(make-stencil-circler 0.1 0.25 ly:text-interface::print)
90     \override Stem.stemlet-length = #0.75
91     \proportionalNotationDuration = #(ly:make-moment 1/16)
92     \remove "SeparatingLineGroupEngraver"
93   }
94   \context {
95     \Staff
96
97     \override VerticalAxisGroup.staff-staff-spacing =
98       #'((basic-distance . 15 )
99         (minimum-distance . 15 )
100         (padding . 0 )
101         (stretchability . 0))
102
103     \override RehearsalMark.X-offset = #1
104     \override RehearsalMark.Y-offset = #4
105     \override VerticalAxisGroup.default-staff-staff-spacing =
106       #'((basic-distance . 16 )
107         (minimum-distance . 16 )
108         (padding . 0 )
109         (stretchability . 0))
110
111     \override TimeSignature.font-size = #2
112     \override TimeSignature.break-align-symbol = #'clef
113     \override TimeSignature.X-offset =
114       #ly:self-alignment-interface::x-aligned-on-self
115     \override TimeSignature.self-alignment-X = #LEFT
116     \override TimeSignature.Y-offset = #9
117     \override TimeSignature.extra-offset = #'(2 . 0)
118     \override TimeSignature.break-visibility = #end-of-line-invisible
119   }
120   \context {
121     \StaffGroup
122     \name "SemiStaffGroup"
123     \consists "SpanBarEngraver"
124     \override SpanBar.stencil =
125       #(lambda (grob)
126         (if (string=? (ly:grob-property grob 'glyph-name) "|")
127             (set! (ly:grob-property grob 'glyph-name) ""))
128         (ly:span-bar::print grob))
129   }
130   \context {
131     \Score
132     \accepts SemiStaffGroup
133   }
134 }
135
136 \score{
137   \new Score
138   <<
139   \new SemiStaffGroup {
140     <<
141     \new Staff \with {
142       instrumentName = "high"
143       shortInstrumentName = "high"
144     }
145     <<
146     \include "includes/cicc.high.ly"
147     >>
148
149     \new Staff \with {
150       instrumentName = "guitar"

```



```

151 shortInstrumentName = "guitar"
152 }
153 <<
154 \include "includes/cicc.guitar.ly"
155 >>
156
157 \new Staff \with {
158   instrumentName = "low"
159   shortInstrumentName = "low"
160 }
161 <<
162 \include "includes/cicc.low.ly"
163 >>
164
165 >>
166 }
167 >>
168
169 \layout{}
170 }

```

cicc_pseudoindents_def.ly

```

1  %%%%%%%%%% HEADER %%%%%%%%%%
2  %
3  % this code was prompted by
4  % https://lists.gnu.org/archive/html/lilypond-user/2019-07/msg00139.html
5  % and offers a pseudoindent hack suitable for general use
6
7  % keywords:
8  % indent short-indent indentation system line
9  % mid-score temporarily arbitrary individual single just only once
10 % coda margin
11 % mouse's tale acrostic mesostic spine
12
13 %%%%%%%%%% PSEUDOINDENT FUNCTIONS %%%%%%%%%%
14
15 % these two functions are for indenting individual systems
16 % - to left-indent a system, apply \pseudoIndent before the music continues
17 % - \pseudoIndents is similar, but lets you also indent on the right
18 % - both provide an option for changing that system's instrument names
19
20 % N.B. these functions
21 % - assume application to non-ragged lines (generally the default)
22 % - include a manual \break to ensure application at line start
23 % - misbehave if called more than once at the same line start
24
25 % the parameters of the (full) pseudoIndents function are:
26 % 1: name-tweaks
27 %   usually omitted; accepts replacement \markup for instrument names
28 %   as an ordered list; starred elements leave their i-names unchanged.
29 % 2: left-indent
30 %   additional left-indentation, in staff-space units; can be negative,
31 %   but avoid a total indentation which implies (unsupported) stretching.
32 % 3: right-indent
33 %   amount of right-indentation, in staff-space units; can be negative.
34 %   - not offered by the (reduced) pseudoIndent function
35
36
37 pseudoIndents = % inline alternative to a new \score, also with right-indent
38 # (define-music-function (parser location name-tweaks left-indent right-indent)
39   ((markup-list? '()) number? number?)
40   (define (warn-stretched p1 p2) (ly:input-warning location (.
41     " pseudoIndents `s `s is stretching staff; expect distorted layout") p1 p2))
42   (let* (
43     (narrowing (+ left-indent right-indent)) ; of staff implied by args
44
45     (set-staffsymbol! (lambda (staffsymbol-grob) ; change staff to new width
46       (let* (
47         (left-bound (ly:spanner-bound staffsymbol-grob LEFT))
48         (left-moment (ly:grob-property left-bound 'when))
49         (capo? (moment<=? left-moment ZERO-MOMENT)) ; in first system of score
50         (layout (ly:grob-layout staffsymbol-grob))
51         (lw (ly:output-def-lookup layout 'line-width)) ; debugging info
52         (indent (ly:output-def-lookup layout (if capo? 'indent 'short-indent)))
53         (old-stil (ly:staff-symbol::print staffsymbol-grob))
54         (staffsymbol-x-ext (ly:stencil-extent old-stil X))
55         ;; >=2.19.16's first system has old-stil already narrowed [2]
56         ;; compensate for this (ie being not pristine) when calculating
57         ;; - old leftmost-x (its value is needed when setting so-called 'width)
58         ;; - the new width and position (via local variable narrowing-)
59         (ss-t (ly:staff-symbol-line-thickness staffsymbol-grob))
60         (pristine? (<= 0 (car staffsymbol-x-ext) ss-t)) ; would expect half
61         (leftmost-x (+ indent (if pristine? 0 narrowing)))
62         (narrowing- (if pristine? narrowing 0)) ; uses 0 if already narrowed
63         (old-width (+ (interval-length staffsymbol-x-ext) ss-t))
64         (new-width (- old-width narrowing-))
65         (new-rightmost-x (+ leftmost-x new-width)) ; and set! this immediately
66         (junk (ly:grob-set-property! staffsymbol-grob 'width new-rightmost-x))
67         (in-situ-stil (ly:staff-symbol::print staffsymbol-grob))
68         (new-stil (ly:stencil-translate-axis in-situ-stil narrowing- X))
69         ; (new-stil (stencil-with-color new-stil red)) ; for when debugging
70         (new-x-ext (ly:stencil-extent new-stil X)))
71         (ly:grob-set-property! staffsymbol-grob 'stencil new-stil)
72         (ly:grob-set-property! staffsymbol-grob 'X-extent new-x-ext)
73       )))
74
75   (set-X-offset! (lambda (margin-grob) ; move grob across to line start
76     (let* (
77       (old (ly:grob-property-data margin-grob 'X-offset))
78       (new (lambda (grob) (+ (if (procedure? old) (old grob) old) narrowing))))
79     (ly:grob-set-property! margin-grob 'X-offset new))))
80
81   (tweak-text! (lambda (i-name-grob mkup) ; tweak both instrumentname texts
82     (if (and (markup? mkup) (not (string=? (markup->string mkup) ""))))
83     (begin
84       (ly:grob-set-property! i-name-grob 'long-text mkup)
85       (ly:grob-set-property! i-name-grob 'text mkup)
86     )) ; else retain existing text
87
88   (install-narrowing (lambda (leftedge-grob) ; on staves, + adapt left margin
89     (define (grob-name x) (assq-ref (ly:grob-property x 'meta) 'name))
90     (let* (
91       (sys (ly:grob-system leftedge-grob))
92       (all-grobs (ly:grob-array->list (ly:grob-object sys 'all-elements)))
93       (grobs-named (lambda (name)
94         (filter (lambda (x) (eq? name (grob-name x))) all-grobs)))
95       (first-leftedge-grob (list-ref (grobs-named 'LeftEdge) 0))
96       (relsys-x-of (lambda (g) (ly:grob-relative-coordinate g sys X)))
97       (leftedge-x (relsys-x-of first-leftedge-grob))
98       (leftedged? (lambda (g) (= (relsys-x-of g) leftedge-x)))
99       (leftedged-ss (filter leftedged? (grobs-named 'StaffSymbol))))
100     (if (eq? leftedge-grob first-leftedge-grob) ; ignore other leftedges [1]

```

```

101 (begin
102   (for-each set-staffsymbol! leftedged-ss)
103   (for-each set-X-offset! (grobs-named 'SystemStartBar))
104   (for-each set-X-offset! (grobs-named 'InstrumentName))
105   (for-each tweak-text! (grobs-named 'InstrumentName) name-tweaks)
106   ))))
107
108 (if (negative? narrowing) (warn-stretched left-indent right-indent))
109 #f % and continue anyway
110 % ensure that these overrides are applied only at begin-of-line
111 \break % (but this does not exclude unsupported multiple application)
112 % give the spacing engine notice regarding the loss of width for music
113 \once \override Score.LeftEdge.X-extent = #(cons narrowing narrowing)
114 % discard line start region of staff and reassemble left-margin elements
115 \once \override Score.LeftEdge.after-line-breaking = #install-narrowing
116 % shift the system to partition the narrowing between left and right
117 \overrideProperty Score.NonMusicalPaperColumn.line-break-system-details
118   X-offset #(- right-indent)
119 % prevent a leftmost barnumber entering a stretched staff
120 \once \override Score.BarNumber.horizon-padding = #(max 1 (- 1 narrowing))
121 #f))
122
123 pseudoIndent = % for changing just left-indent
124 #(define-music-function (parser location name-tweaks left-indent)
125   ((markup-list? '()) number?)
126   #f
127   \pseudoIndents $name-tweaks $left-indent 0
128   #f))
129
130 % [1] versions <2.19.1 can have end-of-line leftedges too
131 % - these were eliminated in issue 3761
132 % [2] versions >=2.19.16: the first system behaves differently from the rest
133 % - a side effect of issue 660 ?

```