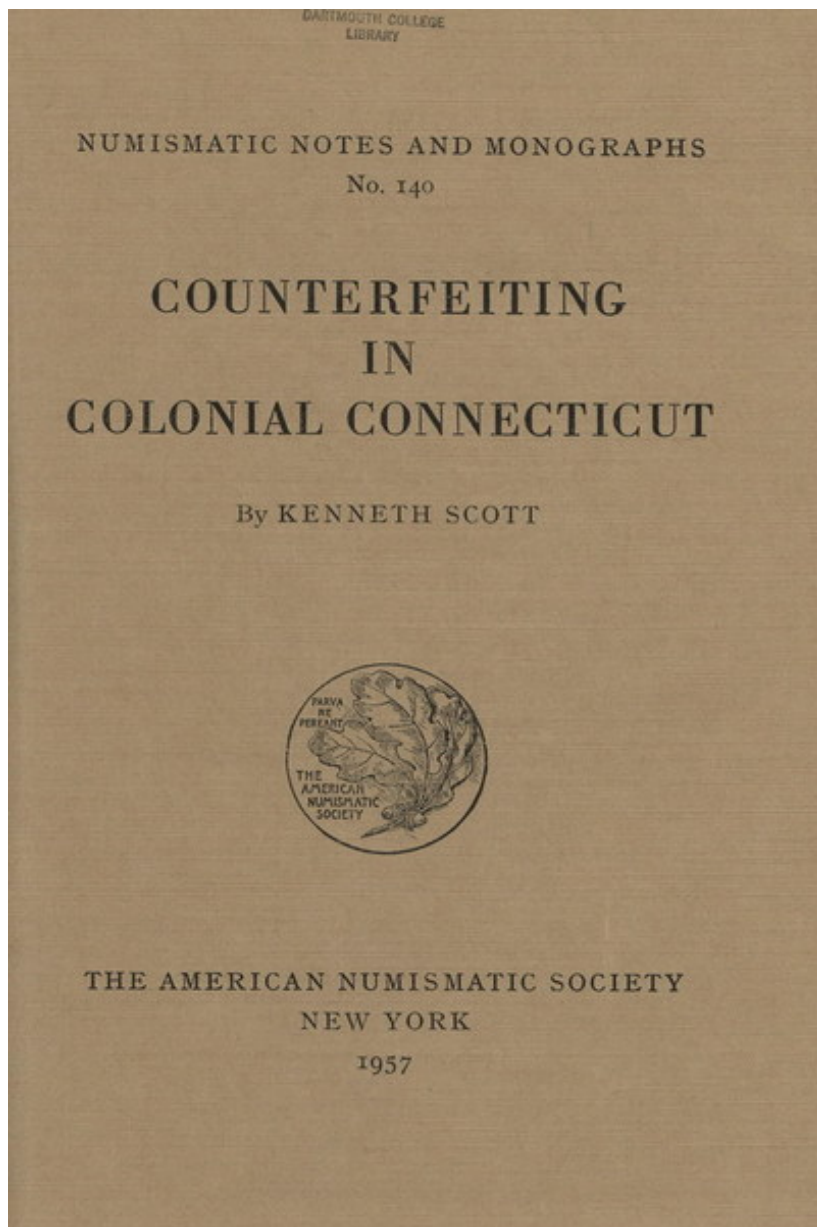


Counterfeiting in Colonial Connecticut
for guitar with low and high accompaniment and reader

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

Preferably played in a dark or dim setting (with minimal 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 - 2020.07.01

general remarks and instructions	1
musical score	5
appendix 1 - excerpts from “Counterfeiting in Colonial Connecticut” by Kenneth Scott	96
appendix 2 - SuperCollider code and Lilypond templates	102

general remarks (to optionally be used as a program note and / or read in 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.

Reading the book compelled me to learn more about the history of counterfeiting. For example, I learned that 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.

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 fitting. 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 counterfeiting as an integral metaphor. The underlying variables in the computer program that generates the piece vary slightly throughout; as if 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 parenthesis):

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 retard.

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 ever 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 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 l.v. 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 such as 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 breath. 8 minutes and 46 seconds in total. It was so clearly murder.

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 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.

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

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 United States constitution is not so holy. The human rights that it outlined did not apply to black people until the 13th amendment was passed. And even then, there were plenty of means of oppression left in tact. 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 on the following page (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.

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.

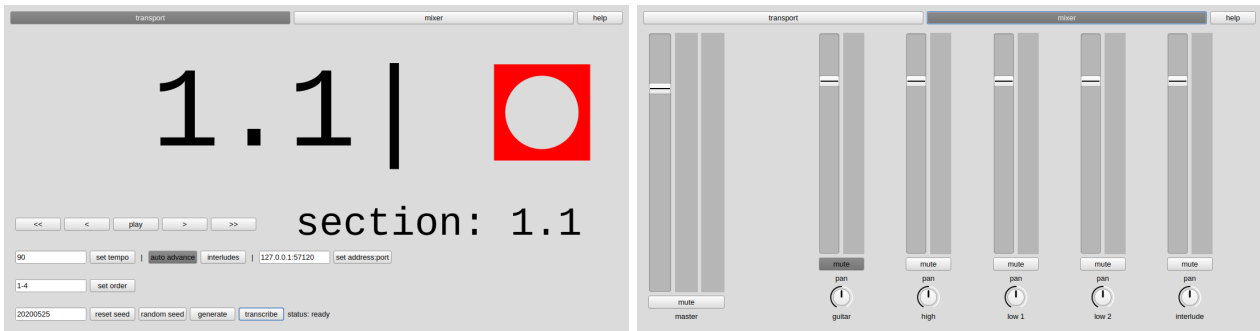
The application source code is appended at the end of this score and downloadable from a git repository at:

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 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

```
1  (//note that this is sensitive to frequency and tremolo rate inputs
2  SynthDef(\interludeTremelo, {arg gate = 0, amp = 1, freq1, freq2, tremRate;
3    var tremeloTrig, trem, freq, sig, feedback, fade;
4    //fast tremelo - note that this can be slower so long as the delaytime of the feedback remains short
5    tremeloTrig = Impulse.kr(tremRate);
6    //tremelo between two notes
7    trem = Select.kr(Stepper.kr(tremeloTrig, 0, 0, 1), [freq1, freq2]);
8    //occasionally tremelo 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), tremeloTrig);
12   //feedback
13   feedback = CombC.ar(sig, 0.2, tremRate.reciprocal, 5);
14   fade = feedback * EnvGen.ar(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(\interludeTremelo, [\gate, 1, \amp, 1, \freq1, freq1, \freq2, freq2, \tremRate, tremRate])
26 )
```

interlude synth code in SuperCollider

I would like to extend a special thanks Alex Bruck and Elliot Simpson. I first encountered the Kenneth Scott compendium at Alex's apartment. Elliot was extremely helpful answering questions about notation that led to the final format of the score of the piece, which is ultimately written for him as a performer in mind.

version generated: 2020.07.01

seed: 20200525

1.1

$\frac{2}{4}$ = approx. 90

(8)

high

gui ar

(9)

V^I1² V^I1¹ V^I1¹

michael winter
(cdmx and gatlinburg, tennessee; 2020)

DRAFT - 2020.07.01

21 (8)

high

guitar

low

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

25 (8)

high

guitar

low

VI⁰ IV⁰ III⁰ II⁰ IV⁰ III⁵ VI⁰ VI⁵ III⁰ VI⁵ III⁵ II⁰ IV⁵ VI⁰ I⁵ I⁶ II⁰ IV⁰ VI⁰

29 (8)

high

guitar

low

I⁰ II⁶ IV³ III⁵ VI⁵ II⁰ IV⁵ VI⁰ 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

DRAFT - 2020.07.01

41 (8)

high

guitar

low

VI⁵ II⁰ IV⁵ I⁶ IV⁰ I⁰ II⁵ IV³

VI⁰

45 (8)

high

guitar

low

III⁵ VI⁵ II⁰ I⁶ I⁶ II⁰ I⁰ II⁵

VI⁰ VI⁰

5.4

49 (8)

high

guitar

low

III⁵ VI⁵ II⁰ IV⁵ I⁸ I⁶ II⁰ II⁵ IV³

VI⁰

4.8

53 (8)

high

guitar

low

III⁵ VI⁵ II⁰ IV⁵ I⁸ I⁶ II⁰ IV⁰

VI⁰

4.3

57 (8)

high

guitar

low

I⁰ II⁵ III⁵ VI⁵ IV⁵ I⁸ I⁶ II⁰ IV⁰ VI⁰

3.8

DRAFT - 2020.07.01

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 II⁶ IV³

73 (8)

high

guitar

low

2.2

III⁵ VI⁵ II⁰ IV⁵ I⁸ II⁰ IV⁰ VI⁰

77 (8)

high

guitar

low

1.6

VI⁰ II⁵ III⁵ VI⁵ II⁰ IV⁵ II⁰ VI⁰

DRAFT - 2020.07.01

high

guitar

low

(81) (89)

IV⁰ VI⁰ IV³ III⁵ VI⁵ IV⁵

1.1

high

guitar

low

(85) (89)

VI⁰ I⁵ I⁶ IV⁰ I⁰ II⁵ IV³ III⁵ VI⁵

0.5

high

guitar

low

(89) (89)

II⁰ IV⁵ VI⁰ I⁵ I⁶ II⁰ I⁰ II⁵ IV³

high

guitar

low

1.3 (89) (89)

I⁴ VI³ II⁰ I⁰ II⁰ I⁰ II⁵ II³ I⁰ VI⁰ V⁶ V⁵ VI⁰ V⁰ VI² V⁰ VI⁰ V⁴ VI⁰ V⁰

0.0

high

guitar

low

(97) (89)

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

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 VI⁰ IV⁰ II⁰ VI⁰ IV⁰ II⁰ VI⁰ I⁰ IV⁰

109 (8)

high

guitar

low

1.4

2/2

0.0

II⁰ V⁰ III³ III¹ IV⁰ VI⁰ I⁰ II⁰ VI⁰ III⁰ I⁰ IV⁰ II⁰ VI⁰

112 (8)

high

guitar

low

IV⁰ II⁰ VI⁰ I⁰ VI⁰ III⁰

116 (8)

high

guitar

low

VI⁰ III⁰ VI⁰ III⁰

DRAFT - 2020.07.01

(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)

(8)

(8)

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

high

guitar

low

(8)

(8)

(8)

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

high

guitar

low

(8)

(8)

(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)

(8)

(8)

II⁰ I⁷ IV⁷ V⁷ II⁰ I⁷ VI⁰ I⁷ IV⁰ VI⁹ III⁰

high

guitar

low

5.6

(8)

(8)

(8)

I⁷ I⁶ II⁰ I⁷ V⁷ II⁰ VI⁰ I⁷

high

guitar

low

20 (8)

4.2

VI⁰ III⁰ I⁷ VI¹⁰ VI⁹ III⁰ I⁷ II⁰ I⁷ IV⁷ V⁷

high

guitar

low

24 (8)

2.8

II⁰ VI⁰ III⁰ IV⁰ VI¹⁰ VI⁹ III⁰ I⁶ II⁰ I⁷

high

guitar

low

28 (8)

IV⁷ V⁷ VI⁰ I⁷ III⁰ I⁷ IV⁰ VI¹⁰ VI⁹ III⁰

high

guitar

low

32 (8)

1.4

I⁶ II⁰ I⁷ IV⁷ V⁷ VI⁰ I⁷ VI⁰

high

guitar

low

36 (8)

2.3

0.0

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 VI⁶ 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 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

60 (8)

high

guitar

low

4.1

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

64 (8)

high

guitar

low

2.8

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

68 (8)

high

guitar

low

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

2.4

72 (8)

high

guitar

low

0.0

V^{*} IV^{*} III^{*} I^{*} II^{*} V^{*} IV^{*} III^{*} I^{*} II^{*} V^{*} IV^{*} I^{*}

76 (8)

high

guitar

low

IV^{*} VI^{*} V^{*} II^{*} III^{*}

21 (8)

high

guitar

low

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

25 (8)

high

guitar

low

IV⁶ VI⁶ VI⁵ III⁷ V⁰ I⁰ IV⁰ III⁰ VI⁰ V⁰ I⁶ IV⁰ VI⁰ III⁷ III⁵ V⁴ I⁶ IV⁰ VI⁰

3.2

6.0

29 (8)

high

guitar

low

III⁷ III⁵ V⁴ I⁶ IV⁰ VI⁰ III⁷ III⁵ V⁴

5.5

33 (8)

high

guitar

low

I⁶ VI⁰ III⁷ III⁵ I⁶ IV⁰ VI⁰ III⁷ V⁴

5.0

4.5

37 (8)

high

guitar

low

I⁶ VI⁰ III⁵ V⁴ I⁶ IV⁰ VI⁰ III⁷ III⁵ V⁴ I⁶

4.0

3.5

3.0

high

guitar

low

(41) (8)

IV⁰ VI⁰ V⁴ I⁶ III⁷ III⁵ I⁶ IV⁰

2.5 2.0

high

guitar

low

(45) (8)

III⁷ III⁵ I⁶ VI⁰ V⁴ I⁶ IV⁰

1.5 1.0

high

guitar

low

(49) (8)

III⁷ V⁴ IV⁰ VI⁰ III⁷ III⁵ I⁶ IV⁰ III⁰ VI⁰

0.5 0.0

3.3

high

guitar

low

(53) (8)

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

high

guitar

low

(57) (8)

II⁶ V⁰ IV⁰ VI³ V⁴ V³ II⁰ V⁰ II⁰ II⁰ IV⁵ IV³ III⁴ V⁰ VI⁰ II⁴ IV⁰ III⁰ V⁰

DRAFT - 2020.07.01

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

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

1.1

73 (8)

high

guitar

low

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

3.4

76 (8)

high

guitar

low

II⁰ VI⁰ I⁰ V⁰ II⁰ VI⁰ V⁰ II⁰

0.0

high (80) (8)

guitar (8)

low (8)

VI° I° V° II°

high (84) (8)

guitar (8)

low (8)

VI° I° V°

high (88) (8)

guitar (8)

low (8)

II°

4.1

high

guitar

low

0.0

5

high

guitar

low

9

high

guitar

low

13

high

guitar

low

17

high

guitar

low

6.0

4.2

high

guitar

low

6.0

IV¹⁰ II⁹ V¹⁰ IV⁰ II⁰ V⁹ III⁰ V⁹ II⁰ I⁹ III⁰ I⁰ III¹⁰ I⁰ III⁹ I⁰ III⁰ I⁰ III⁹ III⁷ I⁰ III⁰ I⁰

III⁷ I⁰ III⁰ I⁷ III⁰ I⁰ III⁷ I⁰ III⁰ I⁷ III⁰ I⁷ III⁰ VI⁰ V⁹ II⁰ IV⁹ III⁷ VI⁰

V⁷ II⁰ IV⁹ IV⁷ III⁰ VI⁹ V⁰ II⁹ IV⁶ III⁰ V⁶ II⁹ IV⁰ III⁰ V⁰ VI⁰ II⁸ III⁶ V⁰ VI⁰ II⁸ II⁶ I⁰

II⁶ I⁰ II⁰ IV⁶ VI⁹ V⁰ IV⁰ II⁰ V⁶ IV⁶ II⁶ V⁰ IV⁶ II⁰ V⁶ IV⁰ II⁶ V⁰ IV⁰ II⁶ VI⁰ VI⁷

II⁰ II⁶ II⁵ II⁶ V⁰ IV⁶ II⁰ V⁶ IV⁰ V⁰ IV⁰

21 (8)

high

guitar

low

5.6

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

25 (8)

high

guitar

low

Chords: IV^0 , Π^6 , V^0 , IV^0 , Π^6 , VI^0 , Π^6 , VI^7 , Π^0

29 (8)

high

guitar

low

5.1

Chords: Π^5 , IV^6 , Π^6 , V^0 , IV^6 , Π^0 , V^6 , IV^0 , Π^6

33 (8)

high

guitar

low

4.7

Chords: V^0 , IV^0 , Π^6 , VI^0 , Π^6 , VI^7 , Π^0 , VI^0 , Π^6 , IV^6 , Π^6 , V^0

37 (8)

high

guitar

low

Chords: IV^6 , IV^0 , Π^6 , V^0 , IV^0 , Π^6 , VI^0 , Π^6 , VI^7 , Π^0 , VI^0 , Π^6

DRAFT - 2020.07.01

high (41) (8)

guitar (8)

low (8)

4.3

II⁶ V⁰ II⁰ V⁶ II⁶ IV⁰ II⁶

high (45) (8)

guitar (8)

low (8)

3.9

VI⁰ II⁶ VI⁷ II⁰ VI⁰ II⁶ II⁵ IV⁶ II⁶

high (49) (8)

guitar (8)

low (8)

V⁰ IV⁶ II⁰ IV⁰ II⁶ V⁰ IV⁰ VI⁰ VI⁷

high (53) (8)

guitar (8)

low (8)

3.4

II⁰ VI⁰ II⁵ IV⁶ IV⁶ II⁰ V⁶ IV⁰ II⁶ V⁰

high (57) (8)

guitar (8)

low (8)

VI⁰ II⁶ II⁶ II⁰ II⁶ II⁵

DRAFT - 2020.07.01

high

guitar

low

3.0

61 (8)

IV⁶ II⁶ V⁰ IV⁶ II⁰ IV⁰ II⁶ V⁰ IV⁰

high

guitar

low

65 (8)

II⁶ VI⁰ II⁶ VI⁷ II⁰ II⁵

high

guitar

low

2.6

69 (8)

II⁶ V⁰ V⁶ IV⁰ II⁶ V⁰ IV⁰ II⁶

high

guitar

low

2.1

73 (8)

VI⁰ VI⁷ VI⁰ II⁶ II⁵ IV⁶ V⁰

high

guitar

low

77 (8)

IV⁶ II⁰ V⁶ IV⁰ II⁶ IV⁰ II⁶

DRAFT - 2020.07.01

high

guitar

low

61 (8)

VI⁰ II⁶ VI⁷ II⁰ VI⁰ II⁶ IV⁶ V⁰

1.7

high

guitar

low

65 (8)

IV⁶ II⁰ V⁶ II⁶ V⁰ IV⁰ II⁶

high

guitar

low

69 (8)

VI⁰ II⁶ VI⁷ II⁰ VI⁰ II⁶ II⁶

high

guitar

low

93 (8)

IV⁶ IV⁶ II⁰ V⁶ IV⁰ II⁶ V⁰ IV⁰ II⁶

1.3

high

guitar

low

97 (8)

VI⁰ II⁶ VI⁷ II⁰ VI⁰ II⁶ IV⁶

0.9

DRAFT - 2020.07.01

101 (8)

high

guitar

low

II⁶ V⁰ IV⁶ II⁰ V⁶ IV⁰ II⁶

105 (8)

high

guitar

low

IV⁰ II⁶ VI⁰ II⁶ VI⁷ VI⁰ II⁶

109 (8)

high

guitar

low

II⁵ IV⁶ II⁶ V⁰ IV⁶ II⁰

0.4

113 (8)

high

guitar

low

IV⁰ II⁶ V⁰ IV⁰ II⁶ VI⁰ II⁶ VI⁷

117 (8)

high

guitar

low

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

4.3

0.0

DRAFT - 2020.07.01

121 (8)

high

guitar

low

(8)

VI⁰ III⁵ VI⁰ VI⁰ VI⁰ II⁰ V⁴ IV⁰ I⁰ # VI⁴ II⁰ V⁰ IV⁰ VI⁴ I⁰ III⁰ II⁰ # VI⁴ I⁶ III⁰ II⁰ VI⁰

125 (8)

high

guitar

low

(8)

I⁰ III⁰ II⁴ II³ VI⁰ I⁰ III⁴ II⁰ I⁵ # III⁰ II⁰ # VI⁴ VI³ III⁴ II⁰ I⁰ V³ II³ IV⁰ VI³ V⁰

129 (8)

high

guitar

low

(8)

II⁰ IV⁴ VI⁰ I⁶ I⁴ V⁰ II² IV² V⁰ VI⁰ II¹ IV⁰ V⁰ VI² II¹ IV² IV¹ III⁰ I⁰ IV⁰

133 (8)

high

guitar

low

(8)

4.5

V⁰ II⁰ III³ I² IV⁰ V³ V² II⁰ III¹ I⁰ IV⁰ V⁰ II⁰ # III⁰ I⁰ IV⁰ V⁰ II⁰ I⁰ VI⁰

137 (8)

high

guitar

low

(8)

2.0

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

DRAFT - 2020.07.01

5.2

20 (8)

high

guitar

low

7.0

Chord progression for measures 20-22:

- Measure 20: Π^0 , I^7 , Π^6 , V^5 , IV^0 , Π^6 , Π^4
- Measure 21: Π^0 , IV^6 , VI^7 , III^0 , VI^0 , IV^5
- Measure 22: Π^0 , IV^6 , VI^7 , III^0 , VI^0 , IV^5

23 (8)

high

guitar

low

6.0

Chord progression for measures 23-26:

- Measure 23: Π^7 , I^0 , VI^7 , $\#VI^6$, IV^5 , Π^0 , I^7 , Π^6 , Π^6 , Π^0 , IV^6 , VI^7
- Measure 24: Π^7 , I^0 , VI^7 , $\#VI^6$, IV^5 , Π^0 , I^7 , Π^6 , Π^6 , Π^0 , IV^6 , VI^7
- Measure 25: Π^7 , I^0 , VI^7 , $\#VI^6$, IV^5 , Π^0 , I^7 , Π^6 , Π^6 , Π^0 , IV^6 , VI^7
- Measure 26: Π^7 , I^0 , VI^7 , $\#VI^6$, IV^5 , Π^0 , I^7 , Π^6 , Π^6 , Π^0 , IV^6 , VI^7

27 (8)

high

guitar

low

Chord progression for measures 27-30:

- Measure 27: III^0 , IV^5 , Π^7 , I^0 , VI^7 , $\#VI^6$, IV^5 , Π^0 , I^7 , Π^6
- Measure 28: III^0 , IV^5 , Π^7 , I^0 , VI^7 , $\#VI^6$, IV^5 , Π^0 , I^7 , Π^6
- Measure 29: III^0 , IV^5 , Π^7 , I^0 , VI^7 , $\#VI^6$, IV^5 , Π^0 , I^7 , Π^6
- Measure 30: III^0 , IV^5 , Π^7 , I^0 , VI^7 , $\#VI^6$, IV^5 , Π^0 , I^7 , Π^6

31 (8)

high

guitar

low

5.0

Chord progression for measures 31-34:

- Measure 31: V^5 , IV^0 , Π^6 , Π^4 , Π^0 , IV^6 , VI^7 , III^0 , IV^5 , VI^0
- Measure 32: V^5 , IV^0 , Π^6 , Π^4 , Π^0 , IV^6 , VI^7 , III^0 , IV^5 , VI^0
- Measure 33: V^5 , IV^0 , Π^6 , Π^4 , Π^0 , IV^6 , VI^7 , III^0 , IV^5 , VI^0
- Measure 34: V^5 , IV^0 , Π^6 , Π^4 , Π^0 , IV^6 , VI^7 , III^0 , IV^5 , VI^0

35 (8)

high

guitar

low

4.0

Chord progression for measures 35-38:

- Measure 35: Π^7 , I^0 , $\#VI^6$, IV^5 , Π^0 , I^7 , Π^6 , IV^0 , Π^6 , Π^4 , Π^0
- Measure 36: Π^7 , I^0 , $\#VI^6$, IV^5 , Π^0 , I^7 , Π^6 , IV^0 , Π^6 , Π^4 , Π^0
- Measure 37: Π^7 , I^0 , $\#VI^6$, IV^5 , Π^0 , I^7 , Π^6 , IV^0 , Π^6 , Π^4 , Π^0
- Measure 38: Π^7 , I^0 , $\#VI^6$, IV^5 , Π^0 , I^7 , Π^6 , IV^0 , Π^6 , Π^4 , Π^0

DRAFT - 2020.07.01

high

guitar

low

(39) (8)

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

(8)

high

guitar

low

(43) (8)

V⁵ IV⁰ II⁶ II⁴ II⁰ IV⁶ VI⁷ III⁰

3.0

(8)

high

guitar

low

(47) (8)

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

2.0

(8)

high

guitar

low

(51) (8)

IV⁶ III⁰ VI⁰ IV⁵ II⁷ I⁰ VI⁷ VI⁶ IV⁵ II⁰ I⁷

(8)

high

guitar

low

(55) (8)

V⁵ IV⁰ II⁴ IV⁶ IV⁵ II⁷ VI⁷ #VI⁶ IV⁵ II⁰ I⁷

1.0

(8)

59 (8)

high

guitar

low

5.3

0.0

63 (8)

high

guitar

low

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

67 (8)

high

guitar

low

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

71 (8)

high

guitar

low

3.1

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

75 (8)

high

guitar

low

2.6

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°

high

guitar

low

21 (8)

high

guitar

low

(8)

III⁰ VI⁰ I⁰ II⁶ III⁶ VI⁰ I⁰ II⁰ III⁰ VI⁶ I⁰ II⁶ III⁰ VI⁶ I⁰ II⁶ I⁰ II⁰ V⁰

25 (8)

high

guitar

low

(8)

I⁶ II⁰ V⁰ I⁶ II⁰ V⁹ I⁶ II⁰ V⁸ I⁰ II⁰ V⁰ I⁵ II⁰ V⁰ IV⁶ VI⁰ V⁰ II⁰ III⁰ IV⁰ I⁰ V⁶ II⁰

29 (8)

high

guitar

low

(8)

V⁵ I⁵ II⁰ VI⁰ V⁵ I⁵ II⁰ VI⁰ V⁵ I⁵ II⁰ VI⁰ V⁵ I⁵ II⁰ V⁵ III⁸ III⁶ I⁵ III⁵ IV⁰ V⁵ III⁶

33 (8)

high

guitar

low

(8)

IV⁵ V⁰ II⁰ VI⁰ III⁰ IV⁰ V⁵ II⁷ VI⁸ III⁰ IV⁰ V⁰ II⁰ VI⁰ I⁰ IV⁵ II⁰ I⁰

37 (8)

high

guitar

low

(8)

IV⁰ II⁷ I⁰ IV⁰ II⁷ I⁰ IV⁰ II⁰ I⁰ IV⁰ II⁵ I⁰ IV⁰ II⁰ I⁰ IV⁵ II⁰ I⁰ IV⁰ II⁵ I⁰ IV⁰ II⁰

41 (8)

high

guitar

low

VI⁰ IV⁰ V⁵ I⁵ III⁵ I⁰ II⁰ IV⁰ I⁰ II⁰ III⁵ V⁰ III⁰ I⁰ III⁰

45 (8)

high

guitar

low

I⁰ III⁵ I⁵ II⁰ VI⁰ #VI⁸ V⁵ III⁰ IV⁰ II⁵ IV⁰ VI⁷ IV⁰ VI⁷ II⁵ II⁴ IV⁰ II⁰ IV⁵ IV⁴ V⁰

49 (8)

high

guitar

low

II⁰ I⁰ #VI⁶ II⁰ VI⁵ I⁰ V⁰ IV⁴ V⁰ I³ II⁶ II⁴ II⁰

6.2

6.0

52 (8)

high

guitar

low

V⁰ II⁰ I⁰ #VI⁶ VI⁵ II⁰ I⁰ V⁰ IV⁴

56 (8)

high

guitar

low

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

5.0

DRAFT - 2020.07.01

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 II⁰ I⁰ VI⁶ II⁰

3.0

high

guitar

low

(72) (8)

VI⁵ II⁰ I⁰ V⁰ IV⁴ V⁰ I³ II⁵ II⁴

2.0

high

guitar

low

(76) (8)

IV⁰ II⁰ IV⁵ V⁰ II⁰ I⁰ VI⁶ VI⁵ I⁰ V⁰

DRAFT - 2020.07.01

high

guitar

low

60 (8)

IV⁴ V⁰ I³ II⁵ IV⁰ II⁰ IV⁴ V⁰ II⁰ I⁰ #VI⁵ II⁰

1.0

high

guitar

low

64 (8)

VI⁵ II⁰ I⁰ V⁰ V⁰ I³ V⁰ I³ III⁴ V⁰ VI⁰

6.3

0.0

high

guitar

low

67 (8)

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

high

guitar

low

91 (8)

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

3.7

high

guitar

low

95 (8)

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 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

(8)

(8)

(8)

III⁰ V⁰ II¹¹ #VI⁸ I⁰ III⁰ V⁰ II⁰ I¹¹ II¹¹ II⁹ I⁰ VI⁰ IV⁰ III⁰ II⁶ I⁹ #VI⁸

5

high

guitar

low

(8)

(8)

(8)

IV⁰ III¹⁰ III⁹ II⁰ I⁰ #VI⁸ IV⁰ III⁸ I⁸ II⁰ #VI⁸ I⁸ II⁸ VI⁸ II⁸ I⁰ III⁰ II⁰ I⁶

9

high

guitar

low

(8)

(8)

(8)

III⁰ II⁰ I⁰ III⁰ II⁰ I⁰ III⁰ II⁰ I⁰ III⁸ II⁸ I⁰ III⁰ II⁸ I⁸ III⁸ II⁰ I⁰

13

high

guitar

low

(8)

(8)

(8)

III⁰ II⁸ I⁸ III⁸ II⁰ I⁰ #VI¹¹ #V⁹ III⁰ II⁰ I⁸ VI⁰ V⁰ III⁸ II⁰ I⁰ #VI⁸ V⁸ III⁰

17

high

guitar

low

(8)

(8)

(8)

II⁸ I⁰ VI⁰ V⁰ IV⁹ I⁶ VI⁷ I⁰ III⁰ I⁰ III⁶ #VI⁶ I⁶ III⁶ VI⁶ I⁰ III⁶ I⁰ #VI⁰ III⁰ VI⁶ II⁰

DRAFT - 2020.07.01

21 (8)

high

guitar

low

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

25 (8)

high

guitar

low

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

28 (8)

high

guitar

low

7.2
2
2

5.0 4.2

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

31 (8)

high

guitar

low

3.3

IV⁷ III⁴ IV⁰ III⁰ VI⁶ IV⁷ III⁴

35 (8)

high

guitar

low

2.5 1.7

IV⁰ III⁰ VI⁶ IV⁷ III⁴ IV⁰ III⁰ VI⁶ IV⁷ III⁴

high

guitar

low

3

7.3

2

0.8

0.0

IV⁰ VI⁶ IV⁷ III⁴ VI⁰ V⁰ II⁰ IV⁰ VI⁰

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 V⁰ I² IV⁰ II⁰ V⁰ VI⁵ VI³ I⁰ V⁰ II⁴

2.0

high

guitar

low

II² VI² I⁰ III⁰ I⁰ III⁰ VI² VI⁰ III⁰ II⁵ I⁵ II⁵ I⁵ II⁵

7.4

0.0

high

guitar

low

(58) (8)

I° II° I° IV° V° III° II°

high

guitar

low

(62) (8)

IV° V° III°

high

guitar

low

(66) (8)

II°

high

guitar

low

The musical score for "The Sound of Silence" by Simon & Garfunkel is presented in a three-staff format. The top staff is for the high voice, the middle staff is for the guitar, and the bottom staff is for the low voice. The key signature is one sharp (F#), and the time signature is 4/4. The score includes a variety of musical notations, including notes, rests, and bar lines. Chord symbols are provided for the guitar part, and the vocal parts are written in a clear, legible font. The score is divided into measures by vertical bar lines, and the overall structure is well-organized and easy to read.

The image displays a musical score for the song "The Sound of Silence" by Simon & Garfunkel. It features three staves: a high vocal line, a guitar accompaniment, and a low vocal line. The key signature is one sharp (F#). The guitar part includes various chord symbols such as III⁰, I⁰, V⁸, II¹¹, II¹⁰, III⁰, VI⁰, V⁷, II¹⁰, II⁸, III⁰, VI⁰, II⁰, IV⁷, III⁰, VI⁷, II⁰, IV⁶, III⁰, and VI⁷. The vocal lines are marked with "high" and "low" and include a measure number 13 in the high vocal line.

The image displays a musical score for the song "The Sound of Silence" by Simon & Garfunkel. It features three staves: a vocal line (labeled "high"), a guitar line, and a bass line (labeled "low"). The guitar line includes chord symbols such as II^0 , IV^6 , III^0 , VI^6 , IV^0 , III^7 , VI^0 , II^0 , IV^5 , III^0 , VI^5 , IV^5 , V^7 , V^5 , I^0 , II^0 , VI^5 , V^0 , and I^7 . The vocal line includes a section marked with a circled "8" and a guitar line with a circled "8". The bass line includes a section marked with a circled "8".

8.2

3

2

2

21 (8)

high

guitar

low

24 (8)

high

guitar

low

28 (8)

high

guitar

low

32 (8)

high

guitar

low

36 (8)

high

guitar

low

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

VI⁰ 7.0

IV⁰ III⁶ VI⁴ VI⁰ 6.2 III⁰ IV⁰ III⁶ VI⁴

VI⁰ 5.4 VI⁰ VI⁴ VI⁰ 4.7

III⁰ IV⁰ III⁶ VI⁰ 3.9 III⁰ IV⁰ III⁶ VI⁴

VI⁰ 3.1 IV⁰ III⁶ III⁰ VI⁰ 2.3 VI⁰ IV⁰

DRAFT - 2020.07.01

40 (8)

high

guitar

low

III⁶ VI⁴ VI⁰ III⁰ VI⁰ III⁶

1.6

44 (8)

high

guitar

low

III⁰ IV⁰ III⁶ VI⁴

0.8

8.3

47 (8)

high

guitar

low

IV⁰ II⁰ V⁰ VI⁰ II⁵ V⁴ VI⁰ II⁴ V⁰ VI⁴ II⁰ V⁴ VI⁴ V⁰ VI⁴ IV⁴ V⁴ IV⁴ II⁰

0.0

51 (8)

high

guitar

low

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

55 (8)

high

guitar

low

VI⁴ II⁰ III⁴ VI⁴ II⁰ IV⁴ I⁰ III⁰ II⁴ IV⁴ I⁰ V⁰ VI⁴ IV⁴ I⁰ V⁰ VI⁰ IV⁰

DRAFT - 2020.07.01

59 (8)

high

guitar

low

(8)

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

63 (8)

high

guitar

low

(8)

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

(8)

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

(8)

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

(8)

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 , $\#II^1$, $\#III^4$, $\#III^2$

79 (8) 8.4

high

guitar

low

(8)

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

0.0

83 (8)

high

guitar

low

(8)

V⁰ I⁰ IV⁰ V⁰

86 (8) 3

high

guitar

low

(8)

I⁰

21 (8)

high

guitar

low

9.2

7.0

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

25 (8)

high

guitar

low

6.4

5.8

II⁰ III⁵ V⁵ I⁰ IV⁶ V⁰ II⁴ IV⁰ II⁰

29 (8)

high

guitar

low

5.3

V⁵ I⁰ V⁰ II⁴ IV⁰ II⁰ V⁵ V⁰ II⁴

33 (8)

high

guitar

low

4.7

4.1

IV⁰ II⁰ III⁵ I⁰ IV⁶ II⁴ IV⁰ II⁰

37 (8)

high

guitar

low

3.5

III⁵ V⁵ I⁰ IV⁶ V⁰ IV⁰ II⁰ III⁵

DRAFT - 2020.07.01

41 (8)

high

guitar

low

2.9

Chord progression: V^5 , I^0 , IV^6 , V^0 , II^4 , IV^0 , II^0 , III^5 , V^5

45 (8)

high

guitar

low

2.3

Chord progression: I^0 , IV^6 , V^0 , II^4 , IV^0 , III^5 , I^0

49 (8)

high

guitar

low

1.8

Chord progression: IV^6 , V^0 , II^4 , IV^0 , II^0 , III^5 , V^5 , I^0

53 (8)

high

guitar

low

1.2

0.6

Chord progression: IV^6 , V^0 , IV^0 , II^0 , III^5 , I^0 , II^4 , IV^0 , II^0

57 (8)

high

guitar

low

Chord progression: III^5 , V^5 , I^0 , IV^6 , V^0 , II^4

DRAFT - 2020.07.01

9.3

61 (8)

high

guitar

low

VI⁰ 0.0 IV⁶ IV⁴ V⁰ II² VI³ IV⁴ V⁰ II² VI⁰ III³ IV² V⁰ III⁰ IV² V⁰ III² IV⁰

65 (8)

high

guitar

low

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

69 (8)

high

guitar

low

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

73 (8)

high

guitar

low

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

4.2

77 (8)

high

guitar

low

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

1.3

9.4

3/2

high

guitar

low

(80)

(81)

I⁰ # III⁰ VI¹ V⁰ II⁰ I⁰ III⁰ VI⁰ V⁰ II⁰ IV⁰ II⁰ VI⁰ II⁰ I⁰

0.0

high

guitar

low

(83)

(84)

II⁰ I⁰ II⁰ I⁰ II⁰ I⁰ II⁰ I⁰ II⁰

high

guitar

low

(87)

(88)

I⁰

The image displays a musical score for the song "The Sound of Silence" by Simon & Garfunkel. It consists of three staves: a vocal line (labeled "high"), a guitar line (labeled "guitar"), and a bass line (labeled "low"). The vocal line features a melody with a final note marked with a "7" (seventh). The guitar line is a complex arrangement of chords and melodic lines, with various chord symbols such as VI^0 , IV^0 , V^9 , V^8 , IV^8 , II^0 , III^0 , V^0 , II^8 , III^0 , II^8 , IV^0 , I^0 , $\sharp VI^8$, III^0 , II^0 , IV^0 , and VI^0 annotated below the notes. The bass line is mostly empty, with a few notes in the first measure. The score is written in a key signature of one sharp (F#) and a 4/4 time signature.

The musical score for "The Sound of Silence" is presented in three staves. The top staff is for the high voice, the middle for guitar, and the bottom for the low voice. The guitar part includes chord symbols such as II^8 , VI^8 , III^0 , VI^0 , III^0 , II^8 , I^0 , III^0 , IV^8 , II^8 , I^0 , III^0 , IV^8 , I^0 , II^0 , V^0 , VI^8 , IV^0 , II^8 , and VI^8 . The score is marked with a key signature of one sharp (F#) and a common time signature (C).

The image shows a musical score for the song "The Sound of Silence" by Simon & Garfunkel. It includes three staves: a high vocal line, a guitar line, and a low vocal line. The guitar part is heavily annotated with chord symbols and fingerings. The high vocal line starts with a circled 13 and an 8, indicating a specific pitch or octave. The low vocal line starts with a circled 20 and an 8. The guitar part includes various chords such as II^8 , VI^0 , VI^8 , IV^0 , III^0 , IV^8 , V^8 , and V^0 , along with fingerings like (8), (9), and (10).

21 (8)

high

guitar

low

10.2

5.0

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

25 (8)

high

guitar

low

4.6

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

29 (8)

high

guitar

low

4.2

3.8

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

33 (8)

high

guitar

low

3.5

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

37 (8)

high

guitar

low

3.1

2.7

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

DRAFT - 2020.07.01

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 , $\#VI^6$, II^0 , VI^0 , V^7 , I^0 , II^0

49 (8)

high

guitar

low

1.5

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

53 (8)

high

guitar

low

1.2

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

57 (8)

high

guitar

low

0.8 0.4

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

DRAFT - 2020.07.01

61 (8)

high

guitar

low

(8)

V⁷ I⁰ II⁰ V⁷ II⁰ III⁹ VI⁶

10.3

65 (8)

high

guitar

low

(8)

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

0.0

(8)

69 (8)

high

guitar

low

(8)

II⁸ VI⁰ II⁷ III⁵ I⁴ II⁷ III⁴ I⁴ II⁶ III⁴ I⁰ II⁰ III⁴ I⁴ II⁴ III⁰ I⁴ II⁴ III⁰ I⁴

73 (8)

high

guitar

low

(8)

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

77 (8)

high

guitar

low

(8)

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

DRAFT - 2020.07.01

high

guitar

low

4.6

high

guitar

low

1.6

high

guitar

low

high

guitar

low

10.4

0.0

high

guitar

low

The musical score for 'The Sound of Silence' is presented in three staves. The top staff is for the vocal part, labeled 'high' and '(8)'. The middle staff is for the guitar, labeled 'guitar' and '(8)', with a capo indicated by a 'C' and a sharp sign. The bottom staff is for the bass part, labeled 'low' and '(8)'. The guitar part includes a solo section with a key signature change to one sharp (F#) and a time signature change to 3/4, marked with a '3' and a sharp sign. The vocal part includes a solo section with a key signature change to one sharp (F#) and a time signature change to 3/4, marked with a '3' and a sharp sign. The bass part includes a solo section with a key signature change to one sharp (F#) and a time signature change to 3/4, marked with a '3' and a sharp sign.

11.1

high

guitar

low

0.0

(8)

(8)

(8)

IV¹⁰ V⁰ II⁰ I¹¹ IV⁰ V⁰ II⁰ I¹⁰ VI¹⁰ VI⁸ III⁹ IV⁰ III⁰ IV⁹ I⁰ VI⁸ II⁹ III⁰ IV⁹ I¹⁰ VI⁰

high

guitar

low

(8)

(8)

(8)

II⁰ I⁰ II⁸ V⁰ IV⁸ III⁰ I⁰ II⁰ V⁰ IV⁸ III⁸ I⁰ II⁸ V⁰ IV⁸ III⁰ V⁰ III⁸ V⁰ III⁸ II⁸ I⁰

high

guitar

low

(8)

(8)

(8)

III⁸ II⁰ I⁰ III⁰ II⁰ I⁰ III⁸ II⁸ I⁰ III⁰ II⁸ IV⁸ V¹⁰ VI⁰ III⁰ II⁰ IV⁷ VI⁰ V⁰

high

guitar

low

(8)

(8)

(8)

VI⁸ VI⁷ II⁰ IV⁷ III⁸ V⁰ VI⁰ II⁷ IV⁰ III⁰ V⁰ VI⁰ II⁰ IV⁷ III⁷ V⁰ VI⁷ IV⁷ III⁷

high

guitar

low

(8)

(8)

(8)

V⁰ VI⁰ II⁷ IV⁷ III⁰ VI⁷ IV⁷ II⁰ VI⁷ IV⁰ II⁰ I⁹ II⁰ I⁰ II⁰ I⁸ II⁰ V⁰ IV⁷ V⁸ II⁶ IV⁰

21 (8)

high

guitar

low

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

24 (8)

high

guitar

low

11.2
2
2

5.0

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

27 (8)

high

guitar

low

4.6 4.2

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

31 (8)

high

guitar

low

3.8

II⁵ II⁴ VI⁵ III⁰ IV⁰ VI⁵ IV⁰ I⁷ II⁵ II⁴

35 (8)

high

guitar

low

3.3 2.9 2.5

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

DRAFT - 2020.07.01

high

guitar

low

(39) (8)

VI⁵ IV⁰ I⁷ II⁶ II⁴ VI⁵ III⁰ IV⁰ VI⁵ IV⁰

2.1

high

guitar

low

(43) (8)

II⁵ II⁴ VI⁵ III⁰ IV⁰ VI⁵ IV⁰ I⁷ II⁵ VI⁵

1.7 1.3

high

guitar

low

(47) (8)

III⁰ IV⁰ VI⁵ I⁷ II⁶ II⁴ VI⁵ III⁰ IV⁰ VI⁵

0.8

high

guitar

low

(51) (8)

IV⁰ I⁷ II⁵ II⁴ VI⁵ III⁰ IV⁰

0.4

high

guitar

low

(54) (8)

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

(8)

V⁵ II⁰ I⁵ IV⁰ III⁰ IV⁴ III⁰ IV⁴ VI⁰ VI⁵ VI⁴ III⁶ IV⁰ VI³ III⁰ IV⁰ III⁰ IV⁰ VI⁰ III⁰ IV⁰ VI⁰ II³ III⁰ IV³

61 (8)

high

guitar

low

(8)

VI⁰ II³ I⁵ V³ III⁶ II⁰ I⁰ V³ III⁰ II⁰ I⁰ V⁰ III⁰ II⁰ I⁵ I⁴ II³ IV³

65 (8)

high

guitar

low

(8)

III⁰ I³ II³ IV³ III⁰ I⁰ II⁰ IV⁰ III⁰ I³ II⁰ IV⁰ III⁰ I⁰ II⁰ IV³ III⁰ I³ II⁰ IV³ III⁰

69 (8)

high

guitar

low

(8)

II³ V⁰ II³ V⁰ VI⁰ V⁰ VI⁰ V⁰ I³ VI³ V³ II³ I³ VI³ V⁰ II⁰ I³ VI⁰ V³

73 (8)

high

guitar

low

(8)

II³ VI³ V³ I⁰ IV⁰ VI³ II⁰ VI³ II⁰ V³ I⁰ III⁰ VI³ IV³ V³ I⁰

77 (8)

high

guitar

low

III⁶ III⁵ VI⁰ IV⁰ V⁰ I⁰ III⁴ VI³ 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 III³ II⁰ IV⁰ VI⁰ V⁰ III⁰ II⁰ IV⁰ V⁰ IV⁰ III⁰ I⁰ II⁰ I⁰ VI⁰

1.5

89 (8)

high

guitar

low

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

high

guitar

low

(97) (8)

(8)

IV° I° II° III°

DRAFT

12.1

2

high

guitar

low

0.0

(8)

(8)

(8)

IV⁰ I¹⁰ II⁰ III⁰ IV⁹ I⁰ II¹⁰ III⁰ IV⁰ I⁹ II⁹ III⁹ I⁰ V⁰ II⁹ V⁹ IV⁰ VI⁰ V⁰

high

guitar

low

(8)

(8)

(8)

I⁹ III⁰ VI¹² VI¹¹ V⁰ I⁰ III⁰ I⁸ III⁰ VI⁰ VI⁰ V⁸ III⁰ II⁰ IV⁸ VI⁰ IV⁰ VI⁹ III⁰ II⁰

high

guitar

low

(8)

(8)

(8)

V⁶ IV⁰ V⁶ III⁰ IV⁰ I⁶ V⁶ III⁰ IV⁰ I⁶ III⁸ IV⁰ I⁰ III⁷ V⁰ I⁰ III⁰ V⁰ I⁶ III⁰ IV⁷

high

guitar

low

(8)

(8)

(8)

V⁰ III⁰ IV⁰ V⁰ III⁰ IV⁶ V⁰ III⁰ IV⁰ V⁰ III⁷ III⁶ IV⁰ V⁰ III⁶ IV⁰ V⁰ III⁶ IV⁰ V⁰ III⁶

high

guitar

low

(8)

(8)

(8)

IV⁰ V⁰ III⁶ IV⁰ V⁰ III⁰ IV⁶ V⁰ III⁰ IV⁰ V⁶ III⁰ IV⁰ V⁶ III⁰ IV⁰ V⁶ III⁰ IV⁰ V⁶ III⁰

DRAFT - 2020.07.01

21 (8)

high

guitar

low

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

25 (8)

high

guitar

low

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

29 (8)

high

guitar

low

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

33 (8)

high

guitar

low

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

12.2

36 (8)

high

guitar

low

V⁰ III⁵ II⁵ V⁰ III⁵ III³ II⁵ III⁵ III³ III⁵ V⁰

5.0 4.4

high (39) (8)

guitar (8) III³ III⁵ III⁵ II⁵ V⁰ III⁵ III³

low (8) 3.9 3.3

high (43) (8)

guitar (8) III⁵ II⁵ V⁰ III⁵ III³ III⁵ II⁵

low (8) 2.8 2.2

high (47) (8)

guitar (8) V⁰ III⁵ III³ III⁵ II⁵ V⁰ III⁵ III³

low (8) 1.7

high (51) (8)

guitar (8) III⁵ II⁵ V⁰ III⁵ II⁵ III⁵

low (8) 1.1 0.6

12.3

high (55) (8)

guitar (8) II⁴ IV⁰ V⁰ I⁴ III³ VI⁰ IV⁰ V⁰ I⁰ III⁰ VI⁶ IV⁵ V⁰ I⁰ III⁰ VI⁰ IV⁴ V⁵

low (8) 0.0

DRAFT - 2020.07.01

59 (8)

high

guitar

low

(8)

VI⁰ IV⁰ V⁰ I⁴ I³ III³ VI⁰ IV⁰ V⁰ I⁰ III³ #VI⁶ IV⁰ V⁰ I⁰ III⁰ VI⁰ IV⁰

63 (8)

high

guitar

low

(8)

V⁴ I⁰ III³ VI⁰ V⁰ VI⁰ I³ V⁰ II⁰ VI⁵ V⁰ VI³ I⁰ #V⁴ II⁰ VI⁰ I⁰ II²

67 (8)

high

guitar

low

(8)

VI⁰ 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

(8)

VI⁰ V⁰ IV² IV⁰ I⁰ VI⁰ V⁰ II⁰ IV⁰ I⁰ III¹ IV⁰ I⁰ V⁰ VI⁰ V⁰ IV⁰ II⁰ I⁰

2.1

12.4

75 (8)

high

guitar

low

(8)

III¹ V⁰ VI⁰ III⁰ V⁰ VI⁰ II⁰ IV⁰ III⁰ V⁰ III⁰ V⁰ III⁰ V⁰ III⁰

0.0

78 (8)

high

guitar

low

(8)

V° III° V° III° V° III° V° III°

82 (8)

high

guitar

low

(8)

II° I° VI°

13.1

high

guitar

low

0.0

5

high

guitar

low

9

high

guitar

low

13

13.2

high

guitar

low

7.0

17

high

guitar

low

The musical score is written for guitar and high/low parts. It consists of five systems of staves. The guitar part is in the middle, with high and low parts above and below it. The score includes measure numbers 13.1, 5, 9, 13, and 17. The guitar part is marked with fret numbers (e.g., III⁰, II⁰, I¹¹, I¹⁰, IV⁰, I⁰, II¹⁰, II⁹, I⁰, IV⁰, V¹⁰, IV⁰, V⁰, I¹⁰, III¹¹, IV⁰, V⁰, I¹⁰, I⁹). The high part is marked with measure numbers 13.1, 5, 9, 13, and 17. The low part is marked with measure numbers 13.1, 5, 9, 13, and 17. The score includes a large 'DRAFT' watermark across the top left.

21 (8)

high

guitar

low

6.3

Chord progression for measures 21-24: V^6 , III^0 , IV^8 , II^8 , III^9 , IV^0 , II^0 , V^0 , II^8 , II^7 , V^7 .

25 (8)

high

guitar

low

5.6

Chord progression for measures 25-28: II^0 , VI^8 , V^7 , V^6 , III^0 , I^8 , II^8 , III^9 , VI^0 .

29 (8)

high

guitar

low

4.9

Chord progression for measures 29-32: II^8 , II^7 , VI^0 , V^7 , VI^8 , V^7 , V^6 , III^0 , IV^8 , I^8 , II^8 , IV^0 .

33 (8)

high

guitar

low

Chord progression for measures 33-36: II^0 , VI^0 , V^0 , II^8 , VI^0 , V^7 , II^0 , VI^8 , V^7 , V^6 .

37 (8)

high

guitar

low

4.2

Chord progression for measures 37-40: III^0 , IV^8 , I^8 , II^8 , III^9 , II^0 , VI^0 , V^0 , II^7 , VI^0 .

DRAFT - 2020.07.01

high

guitar

low

3.5

Chord symbols: VI^8 , V^6 , III^0 , I^8 , IV^0 , II^0 , VI^0 , V^0 , II^8 , VI^0 , V^7

high

guitar

low

2.8

Chord symbols: II^0 , VI^8 , V^7 , V^6 , III^0 , IV^8 , I^8 , II^8 , III^9 , IV^0

high

guitar

low

Chord symbols: II^0 , VI^0 , V^0 , II^8 , II^7 , VI^0 , V^7 , II^0 , VI^8 , V^7

high

guitar

low

2.1

Chord symbols: V^6 , I^8 , II^8 , II^0 , VI^0 , II^8 , VI^0 , V^7 , II^0 , VI^8

high

guitar

low

1.4

Chord symbols: V^7 , V^6 , IV^8 , I^8 , II^8 , III^9 , IV^0 , VI^0 , V^0

DRAFT - 2020.07.01

high

guitar

low

(61) (8)

II⁶ II⁷ VI⁰ V⁷ II⁰ #VI⁸ V⁷ #V⁶ III⁰ 0.7

high

guitar

low

(65) (8)

IV⁸ I⁶ II⁶ III⁹ II⁰ VI⁰ II⁸ II⁷ II⁰ #VI⁸ VI⁰ VI⁸

high

guitar

low

(69) (8)

13.3

V⁷ V⁶ II⁶ VI⁰ V⁶ II⁰ VI⁷ III⁰ IV⁸ IV⁶ III⁹ II⁰ V⁰ IV⁰ III⁹ II⁰ V⁶ IV⁰ 0.0

high

guitar

low

(73) (8)

II⁰ IV⁶ II⁰ IV⁶ I⁸ VI⁷ VI⁶ IV⁰ I⁰ VI⁰ IV⁰ II⁶ III⁸ III⁸ VI⁰ V⁰ III⁶ VI⁶ VI⁵ V⁰ IV⁶ VI⁰

high

guitar

low

(77) (8)

VI⁵ III⁰ II⁰ I⁶ II⁵ III⁰ IV⁶ IV⁵ II⁴ III⁵ IV⁵ IV³ III⁴ IV⁰ III³ IV³ V⁶ V⁵ I⁶ I⁵ III⁰ VI⁰

The image displays a musical score for the song "The Sound of Silence" by Simon & Garfunkel. It features three staves: a vocal line (labeled "high"), a guitar line (labeled "guitar"), and a low line (labeled "low"). The vocal line includes a circled measure number "85" and a rehearsal mark "(8)". The guitar line includes a circled measure number "(8)". The low line includes a circled measure number "(8)". The guitar line is annotated with chord symbols: I², I¹, III³, III¹, VI², V³, V¹, I¹, III¹, VI¹, V⁰, I⁰, III¹, VI¹, V⁰, I⁰, and V⁰. The low line contains rests in the first four measures.

The musical score is presented in three staves. The top staff is for the voice, with a treble clef and a key signature of one sharp (F#). The melody is written in a high register, with notes often beamed together. The middle staff is for the guitar, with a treble clef and a key signature of one sharp. It features a complex, melodic line with many beamed eighth and sixteenth notes. The bottom staff is for the bass, with a bass clef and a key signature of one sharp. It contains a simple, rhythmic line with many rests. Chord symbols are written below the guitar staff, corresponding to the chords played. The score is divided into four measures by vertical bar lines.

The image shows a musical score for the song "The Sound of Silence" by Simon & Garfunkel. It consists of three staves: a vocal line (labeled "high"), a guitar line (labeled "guitar"), and a bass line (labeled "low"). The vocal line starts with a treble clef and a key signature of one flat (B-flat). The guitar line starts with a treble clef and a key signature of one flat. The bass line starts with a bass clef and a key signature of one flat. The guitar line includes chord symbols: II^0 , V^0 , III^1 , III^0 , II^0 , V^0 , III^0 , II^3 , II^1 , V^0 , III^0 , II^0 , V^0 , II^0 , IV^0 , II^0 , III^0 , IV^0 , V^0 , and VI^0 . The bass line includes a time signature of 4.9. The score is for a 12-string guitar, as indicated by the "12" in the top right corner.

high

guitar

low

101 (8)

high

guitar

low

2.4

13.4

105 (8)

high

guitar

low

0.0

109 (8)

high

guitar

low

The musical score for "The Sound of Silence" by Simon & Garfunkel is presented in three staves. The top staff is for the vocal part, the middle for guitar, and the bottom for low bass. The guitar part includes chord symbols: Π^9 , III^0 , V^0 , VI^0 , I^{11} , I^{10} , Π^9 , III^0 , V^0 , VI^0 , I^9 , V^0 , I^9 , IV^{11} , VI^0 , IV^{10} , I^9 , Π^9 , V^0 , and IV^9 . The vocal part includes a circled 5 and a circled 8.

The image displays a musical score for the song "The Sound of Silence" by Simon & Garfunkel. The score is written for three parts: high voice, guitar, and low voice. The key signature is one sharp (F#), and the time signature is 4/4. The guitar part includes chord symbols and fingering numbers. The high voice part includes a circled number 13 at the beginning. The low voice part includes a circled number 8 at the beginning.

[illegible]

21 (8)

high

guitar

low

Chord progression for measures 21-24: I⁰, V⁵, II⁰, V⁰, II⁶, II⁵, V⁰, II⁴, V⁰, VI⁰, IV⁵, I⁰, V⁵, V⁴, VI⁰, IV⁰, I⁰.

25 (8)

high

guitar

low

14.2

7.0

Chord progression for measures 25-28: V⁴, II⁰, IV⁰, I⁰, III⁵, IV⁰, II⁴, VI⁵, II³, V⁴, VI⁰, IV⁰, I⁰, V⁴, II⁰.

29 (8)

high

guitar

low

5.6

Chord progression for measures 29-32: IV⁰, I⁰, III⁵, IV⁰, II⁴, VI⁵, II³, VI⁰, I⁰, V⁴, II⁰.

33 (8)

high

guitar

low

4.2

Chord progression for measures 33-36: IV⁰, I⁰, IV⁰, VI⁵, II³, IV⁰, I⁰, V⁴, II⁰.

37 (8)

high

guitar

low

2.8

Chord progression for measures 37-40: IV⁰, III⁵, IV⁰, II⁴, VI⁵, II³, V⁴, IV⁰, I⁰, V⁴, II⁰.

DRAFT - 2020.07.01

41 (8)

high

guitar

low

IV⁰ I⁰ III⁵ II³ V⁴ VI⁰ IV⁰ I⁰ V⁴

1.4

45 (8)

high

guitar

low

II⁰ IV⁰ I⁰ III⁵ IV⁰ VI⁵ II³ V⁰ III⁴ VI⁰ I⁰ III⁰

14.3

0.0

49 (8)

high

guitar

low

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

53 (8)

high

guitar

low

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

57 (8)

high

guitar

low

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

61 (8)

high

guitar

low

(8)

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

65 (8)

high

guitar

low

(8)

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

4.4

(8)

69 (8)

high

guitar

low

(8)

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

2.7

(8)

73 (8)

high

guitar

low

(8)

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

14.4

0.0

(8)

77 (8)

high

guitar

low

(8)

V⁰ III⁰ I⁰ II⁰ V⁰ III⁰ IV⁰ III⁰ V⁰

high

guitar

low

(61) (8)

(6) (8)

IV°

VI°

III°

The image displays a musical score for the song "The Sound of Silence" by Simon & Garfunkel. The score is arranged for three parts: high voice, guitar, and low voice (bass). The key signature is one sharp (F#), and the time signature is 4/4. The score begins with a rehearsal mark (17) and a tempo marking of (8). The high voice part features a melodic line with a long note on the first staff. The guitar part provides harmonic support with various chords, including VI⁷, V⁸, V⁷, III⁷, I⁰, VI⁰, V⁰, III⁷, IV⁰, VI⁷, II⁰, III⁷, V⁰, IV⁰, VI⁷, II⁸, III⁷, V⁰, IV⁸, and VI⁷. The low voice part is mostly silent, indicated by a long note on the first staff.

21 (8)

high

guitar

low

(8)

II⁰ III⁰ V⁰ IV⁷ VI⁰ II⁰ III⁷ V⁰ IV⁰ VI⁷ II⁷ III⁷ V⁷ IV⁰ II⁰ III⁰

25 (8)

high

guitar

low

(8)

V⁰ IV⁰ VI⁷ II⁷ III⁷ V⁷ IV⁷ VI⁷ II⁷ IV⁰ I⁸ VI⁰ II⁰ V⁵ IV⁶ III⁰ IV⁰ III⁷ VI⁰ III⁰ VI⁰

29 (8)

high

guitar

low

(8)

IV⁵ V⁵ I⁰ III⁷ VI⁵ IV⁰ V⁰ I⁰ III⁰ IV⁰ V⁵ I⁶ III⁷ VI⁴ IV⁴ V⁰ I⁰ III⁰ I⁰ V⁰

33 (8)

high

guitar

low

(8)

IV⁰ III⁶ I⁰ V⁰ IV⁴ V⁰ IV⁴ II⁶ III⁶ VI³

15.2

2

7.0

VI⁴ IV⁴ V⁰

36 (8)

high

guitar

low

(8)

I⁰ I⁰ V⁰ IV⁰ III⁶ V⁰ IV⁴ IV⁴ III⁶ VI³

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

DRAFT - 2020.07.01

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

(76) (8)

15.3

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 I³ III⁰ V⁰ I⁰ III⁰ V⁰ I³ III⁰ V³ I⁰

high

guitar

low

(84) (8)

III⁰ V⁰ I⁰ VI³ 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 III⁰ V⁰ I⁰

high

guitar

low

(96) (8)

15.4

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

0.0

100 (8)

high

guitar

low

V° IV° I° III°

103 (8)

high

guitar

low

VI°

The image displays a musical score for the song "The Sound of Silence" by Simon & Garfunkel. It consists of three staves: a high vocal line, a guitar accompaniment, and a low vocal line. The high vocal line begins with a treble clef and a key signature of one sharp (F#), with a circled 5 indicating the starting pitch. The guitar staff features a treble clef, a key signature of one sharp, and a circled 6 indicating the starting fret. Below the guitar staff, chord symbols are provided for each measure: IV⁸, I⁰, III⁰, VI⁰, V⁸, IV⁸, I⁰, III⁰, VI⁰, V⁰, IV⁰, III⁰, I⁹, VI⁹, VI⁷, II⁸, IV⁰, III⁰, I⁰, VI⁷, and #VI⁶. The low vocal line starts with a bass clef and a circled 8 indicating the starting pitch. The score is divided into four measures by vertical bar lines.

high

guitar

low

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

The image displays a musical score for the song "The Sound of Silence" by Simon & Garfunkel. It consists of three staves: a high vocal line, a guitar line, and a low vocal line. The guitar part is heavily annotated with chords and chord progressions. The chords are labeled as follows: IV⁶, #VI⁶, III⁷, I⁰, II⁰, #III⁰, I⁸, V⁰, I⁰, IV⁰, VI⁵, I⁰, IV⁰, VI⁰, V⁰, III⁰, VI⁵, II⁵, IV⁰, and I⁸. The score is marked with a key signature of one sharp (F#) and a time signature of 4/4. The tempo is indicated as "Moderato". The score is numbered 13 and includes a section marked (8).

The image shows a musical score for the song "The Sound of Silence" by Simon & Garfunkel. It features three staves: a high vocal line, a guitar line, and a low vocal line. The guitar part is heavily annotated with chord symbols and fingerings. The high vocal line includes a circled measure number 17 and a breath mark. The low vocal line has a circled measure number 20. The guitar staff includes a key signature of one sharp (F#) and a common time signature (C). Chord symbols such as I⁶, V⁰, VI⁰, II⁵, IV⁰, I⁰, III⁰, and VI⁵ are placed below the guitar staff. Fingerings like (8) and (7) are indicated for specific notes. The high vocal line has a circled measure number 17 and a breath mark. The low vocal line has a circled measure number 20.

21 (8)

high

guitar

low

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

16.2

25 (8)

high

guitar

low

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

5.0

29 (8)

high

guitar

low

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

4.6

33 (8)

high

guitar

low

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

37 (8)

high

guitar

low

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

4.2

DRAFT - 2020.07.01

high

guitar

low

(41) (8)

III⁰ V⁰ VI⁰ III⁰ IV⁰ V⁸ V⁷ VI⁰ IV⁴ II⁰

3.8

high

guitar

low

(45) (8)

VI⁰ III⁵ I⁰ V⁰ I⁵ V⁰ III⁰ V⁰ VI⁰ III⁰

high

guitar

low

(49) (8)

IV⁰ V⁸ VI⁰ IV⁴ II⁰ VI⁰ IV⁵ III⁵ I⁰

3.3

high

guitar

low

(53) (8)

V⁰ III⁵ I⁵ V⁰ III⁰ VI⁰ III⁰ IV⁰

high

guitar

low

(57) (8)

V⁸ V⁷ VI⁰ III⁰ IV⁴ IV⁵ III⁵ I⁰

2.9

DRAFT - 2020.07.01

61 (8)

high

guitar

low

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

65 (8)

high

guitar

low

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

2.5

69 (8)

high

guitar

low

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

2.1

73 (8)

high

guitar

low

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

77 (8)

high

guitar

low

Chords: V^7 , III^0 , VI^0 , III^5 , I^0 , V^0 , III^5

1.7 1.7

high

guitar

low

(81) (8)

I^5 III^0 V^0 VI^0 IV^0 V^8 V^7 VI^0 IV^4

high

guitar

low

(85) (8)

II^0 VI^0 III^5 I^0 V^0 III^5 I^5 III^0

1.3

high

guitar

low

(89) (8)

V^0 VI^0 IV^0 V^8 V^7 VI^0 III^0 II^0 VI^0

0.8

high

guitar

low

(93) (8)

III^5 I^0 V^0 III^5 I^5 V^0 III^0 V^0 VI^0

high

guitar

low

(97) (8)

III^0 IV^0 V^8 V^7 VI^0 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

3/2

109 (8)

high

guitar

low

0.0

I⁰ V⁷ III⁰ VI⁰ II⁵ II⁴ IV⁰ V⁷ III⁴ VI⁰

112 (8)

high

guitar

low

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

116 (8)

high

guitar

low

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

DRAFT - 2020.07.01

120 (8)

high

guitar

low

VI⁴ V⁰ I⁶ IV⁰ V⁰ III³ IV⁰ VI³ VI⁰ V⁰ VI⁰ V⁰ IV⁰ VI⁰ V⁰ IV⁴ IV³ VI³ V⁰ IV³ I⁶ I³

124 (8)

high

guitar

low

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

128 (8)

high

guitar

low

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

132 (8)

high

guitar

low

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

136 (8)

high

guitar

low

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

144 (8)

high

guitar

low

3

7

16.4

147 (8)

high

guitar

low

0.0

151 (8)

high

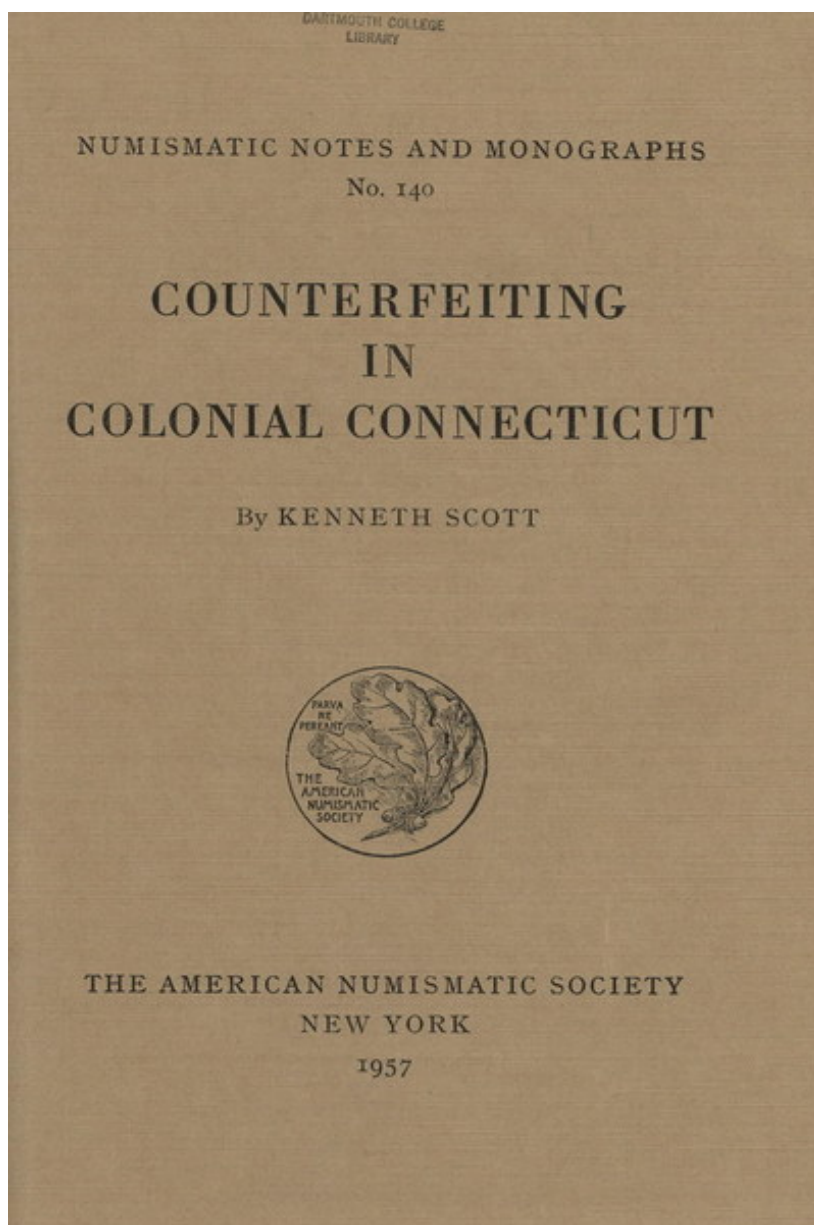
guitar

low

appendix 1 - excerpts from “Counterfeiting in Colonial Connecticut” by Kenneth Scott

reprinted from <http://numismatics.org/digitallibrary/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 - 2020.07.01

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 Read 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 rhead 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 ----transport tab
6 The play button will always start from the beginning of the current section.
7
8 The transport buttons allow you to advance by subsection (<,>) and section (<<,>>).
9
10 Tempo change will only go into effect once "set tempo" button is pressed
11
12 Turning the "auto advance" button on will automatically move from one subsection to the other.
13
14 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
15 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
18 automatic advance between the ultimate subsection of a section and the first subsection of the following section. That is, the performer will have
19 manually advance after the last subsection of each section. The interlude synth will automatically fade out once a new section has been triggered.
20
21 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
22 section 10 and then from section 1 to section 3. This will only go into effect once the "set order" button is pressed.
23
24 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
25 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
26 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
27 the file must be rendered from that location as it depends on files in that folder and the "includes" subfolder.
28
29 ----mixer tab
30 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.
31 The low accompaniment has two separate tracks in case a performer cannot play both the notes.
32 */
```

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             //probability 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
35 strings = (0..5);
36 state = 6.collect({[0, 1].wchoose([2, 1].normalizeSum)}); //fretted or not
37 lastStrings = [nil, nil];
38 position = 6.collect({[10 + 3.rand]}); //which frets
39 dur = setDur.value(modelInitSeq[0][5]);
40 openStringCount = 0;
41 landingCount = 0; //for extending section landing on open strings
42 sectionCount = 0;
43
44 res = []; //notes before the more static repetitions are put in
45
46 //----run routine and create template sequence
47 //----number of sections must be even - generate 16 by default
48 ({sectionCount < 16}).while({
49     var alterPattern, penultimatePos, lastFrettedString, forceUltimateDescent;
50
51     //alter string pattern or not
52     penultimatePos = (position.sign.sum == 1);
53     if(penultimatePos, {lastFrettedString = position.sign.indexOf(1)});
54     forceUltimateDescent = penultimatePos && strings.includes(lastFrettedString).not;
55     alterPattern = [true, forceUltimateDescent].wchoose([1, modelInitSeq[sectionCount][0].normalizeSum);
56     if(alterPattern, {
57         //var lastFrettedString;
58         strings = (0..5).scramble[..(4.rand + 1)];
59         //keep selecting until you have the final string
60         while({forceUltimateDescent && strings.includes(lastFrettedString).not}, {
61             strings = (0..5).scramble[..(4.rand + 1)];
62         });
63         //rotate if a note gets repeated
64         if(lastStrings.last == strings.first, {strings = strings.rotate});
65         lastStrings = strings;
66     });
67
68     //iterate through the strings
69     strings.do({arg string, stringIndex;
70         var alterPos;
71
72         //alter fret if fretted and keeping hand in similar position
73         alterPos = (position[string] * state[string]) > 0;
74         alterPos = alterPos && (state[string] == 1); //isFretted
75         alterPos = alterPos && (position[string] > (position.maxItem - 3));
76         alterPos = [alterPos, false].wchoose([modelInitSeq[sectionCount][1], 1].normalizeSum);
77         if(alterPos, {
78             var walkDown, stepLimit;
79
80             //walk down or not
81             walkDown = [true, false].wchoose([1, modelInitSeq[sectionCount][2]].normalizeSum);
82             if(walkDown, {
83                 res = res.add([string, state[string] * position[string], dur, position.deepCopy]);
84             });
85
86             //make sure a hand position is not too wide
87             stepLimit = (position.maxItem - (position[string] - 2)) != 4;
88             if(stepLimit.not, {
89                 position[string] = (position[string] - 1).clip(0, 12);
90             });
91             if(stepLimit, {
92                 position[string] = (position[string] - [1, 2].choose).clip(0, 12);
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 instad 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 // If chord randomly choose one of the notes
264 if(guitarSeq.last[2] == 0, {
265   arg toAdd = [];
266   toAdd = toAdd.add(guitarSeq.pop);
267   toAdd = toAdd.add(guitarSeq.pop);
268   toAdd[0][2] = toAdd[1][2];
269   toAdd[1][3] = toAdd[0][3];
270   toAdd = toAdd.choose;
271   guitarSeq = guitarSeq.add(toAdd);
272 });
273
274 if(index < (reps - 1), {
275   guitarSeq = extendToBeat.value(guitarSeq, 4)[0];
276 }, {
277   guitarSeq = extendToBeat.value(guitarSeq, 8)[0];
278 });
279
280 insertTS.value(guitarSeq, timeStampSectionStart, 0, true);
281
282 timeStampSection = 0;
283 timeStampSectionStart = guitarSeq.slice(nil, 2).sum;
284 lastDur = initSeq[index + 1][2];
285 state = 2;
286 });
287
288 [guitarSeq, sectionSeq, timeSigInsSeq]
289 };
290
291 // add the high note part
292 finalizeAccompHigh = {arg sectionSeq;
293   var accompHighSeq, timeStamp, subSecType, modelAccomp;
294   accompHighSeq = [];
295   timeStamp = 0;
296   subSecType = 0;
297
298   modelAccomp = sectionSeq.size.collect({
299     [
300       //short probability
301       1.5 + 0.5.rand2,
302       //rest probability
303       3 + 1.0.rand2,
304       //short note average
305       20 + 5.rand2,
306       //short note range
307       5 + 3.rand2,
308       //long note average
309       50 + 10.rand2,
310       //long note range
311       10 + 5.rand2,
312       //rest average
313       40 + 10.rand2,
314       //rest range
315       5 + 5.rand2,
316       //internote space (short rest)
317       6.rand
318     ]
319   });
320
321   sectionSeq.do({arg subSecData, subSecIndex;
322     var subSecEnd, freq, noRestCount, shortCount;
323     subSecEnd = subSecData[0];
324     freq = if(subSecIndex.even, {62.midicps * 8}, {62.midicps * 8 * 6/5});
325     if(subSecData.last, {subSecType = ((subSecType + 1) % 2)});
326     noRestCount = 0;
327     shortCount = 0;
328     while({timeStamp < subSecEnd}, {
329       var dur, sus, isShort, insertRest;
330
331       isShort = case
332       {shortCount == 0} {true}
333       {shortCount < 3} {[true, false].wchoose([modelAccomp[subSecIndex][0], 1].normalizeSum)}
334       {true} {false};
335
336       insertRest = [true, noRestCount > 3].wchoose([modelAccomp[subSecIndex][1], 1].normalizeSum);
337
338       if(isShort, {
339         sus = (modelAccomp[subSecIndex][2] + modelAccomp[subSecIndex][3].rand2).round(2);
340         shortCount = shortCount + 1;
341       }, {
342         sus = (modelAccomp[subSecIndex][4] + modelAccomp[subSecIndex][5].rand2).round(2);
343         shortCount = 0;
344       });
345
346       if(insertRest, {
347         dur = sus + (modelAccomp[subSecIndex][6] + modelAccomp[subSecIndex][7].rand2).round(2);
348         noRestCount = 0;
349       }, {
350         dur = sus + 2 + modelAccomp[subSecIndex][8].rand.round(2);
351         noRestCount = noRestCount + 1;
352       });
353
354       if((timeStamp + dur) < subSecEnd, {
355         accompHighSeq = accompHighSeq.add([freq, dur, sus.clip(0, dur)]);
356       }, {
357         var remainder;
358         remainder = (subSecEnd - timeStamp);
359         sus = if(remainder > 10, {(remainder - 10).rand + 8}.round(2), {0});
360         dur = (remainder + 10.rand).clip(2, 1000).round(2);
361         accompHighSeq = accompHighSeq.add([freq, dur, sus]);
362       });
363       timeStamp = timeStamp + dur;
364     });
365   });
366   accompHighSeq
367 };
368
369 // add the low note part
370 finalizeAccompLow = {arg guitarSeq, sectionSeq;
371   var accompLowSeq, durAccum, lastTrigVal;
372   accompLowSeq = [];
373   durAccum = 0;
374   lastTrigVal = 0;
375   guitarSeq.do({arg item, i;
376     var dur, trig, freq1, freq2, finalDur;
377     dur = item[2];
378     trig = item.last;
379     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 )

```

sonifier.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
12 SynthDef(\`masterPlayerControl ++ `hash, {
13   arg sel = 0,
14   masterVol = 1, masterMute = 1,
15   guitarVol = 1, guitarPan = 0, guitarMute = 0,
16   accompHighVol = 1, accompHighPan = 0, accompHighMute = 1,
17   accompLowLowerVol = 1, accompLowLowerPan = 0, accompLowLowerMute = 1,
18   accompLowUpperVol = 1, accompLowUpperPan = 0, accompLowUpperMute = 1,
19   interludeVol = 1, interludePan = 0, interludeMute = 1;
20   var guitarSig, accompHighSig, accompLowLowerSig, accompLowUpperSig, interludeSig,
21   guitarSigPanned, accompHighSigPanned, accompLowLowerSigPanned, accompLowUpperSigPanned, interludeSigPanned,
22   masterSig, imp;
23
24   guitarSig = In.ar(`guitarBus) * guitarVol;
25   accompHighSig = In.ar(`accompHighBus) * accompHighVol;
26   accompLowLowerSig = Mix.ar(
27     [
28       In.ar(`accompLowLowerBusA) * EnvGen.kr(Env.asr(0.001, 1, 0.1), (sel + 1) % 2),
29       In.ar(`accompLowLowerBusB) * EnvGen.kr(Env.asr(0.001, 1, 0.1), sel)
30     ]
31   ) * accompLowLowerVol;
32   accompLowUpperSig = Mix.ar(
33     [
34       In.ar(`accompLowUpperBusA) * EnvGen.kr(Env.asr(0.001, 1, 0.1), (sel + 1) % 2),
35       In.ar(`accompLowUpperBusB) * EnvGen.kr(Env.asr(0.001, 1, 0.1), sel)
36     ]
37   ) * accompLowUpperVol;
38   interludeSig = In.ar(`interludeTremoloBus) * interludeVol;
39
40   guitarSigPanned = Pan2.ar(guitarSig * guitarMute, guitarPan);
41   accompHighSigPanned = Pan2.ar(accompHighSig * accompHighMute, accompHighPan);
42   accompLowLowerSigPanned = Pan2.ar(accompLowLowerSig * accompLowLowerMute, accompLowLowerPan);
43   accompLowUpperSigPanned = Pan2.ar(accompLowUpperSig * accompLowUpperMute, accompLowUpperPan);
44   interludeSigPanned = Pan2.ar(interludeSig * interludeMute, interludePan);
45   masterSig = Mix.ar(
46     [
47       guitarSigPanned,
48       accompHighSigPanned,
49       accompLowLowerSigPanned,
50       accompLowUpperSigPanned,
51       interludeSigPanned
52     ]) * masterVol * masterMute;
53
54   Out.ar(0, masterSig);
55
56   imp = Impulse.kr(10);
57   SendReply.kr(imp,
58     `masterLevels' ++ `hash,
59     values: [Amplitude.kr(masterSig)]);
60   SendReply.kr(imp,
61     `trackLevels' ++ `hash,
62     values:

```

```

63 [
64   Amplitude.kr(guitarSig), Amplitude.kr(accompHighSig),
65   Amplitude.kr(accompLowLowerSig), Amplitude.kr(accompLowUpperSig),
66   Amplitude.kr(interludeSig)
67 ]
68 );
69 }).add;
70
71
72 SynthDef(\transport ++ ^hash, {arg measure = 0, beat = 0, gate = 1, dur = 1;
73   SendReply.kr(impulse.kr(0) * (measure > 0) * (beat > 0), '/measureClock' ++ ^hash, values: [measure, beat]);
74   SendReply.kr(impulse.kr(0) * (measure < 1) * (beat < 1), '/nextSubsection' ++ ^hash);
75   EnvGen.kr(Env.sine(dur), gate, doneAction: 2);
76 }).add;
77
78
79 //----karplus
80 SynthDef(\karplus ++ ^hash, {arg freq, gate = 1, amp = 0.5, bus;
81   Out.ar(bus,
82     Pluck.ar(WhiteNoise.ar(0.1), Impulse.kr(0), 220.reciprocal, freq.reciprocal, 10, coef:0) *
83     Linen.kr(gate, doneAction: 2) * amp)
84   }).add;
85
86
87 //----accompaniment
88 SynthDef(\accompBass ++ ^hash, {arg freq1 = 100, freq2 = 100, gate = 1, amp = 0.5, busLower, busUpper, cutoff = 0;
89   var env, lower, upper;
90   env = EnvGen.kr(Env.perc(0.1, 10, level: amp), Impulse.kr(0) + Changed.kr(freq2));
91   lower = SinOsc.ar(freq1, 0, 0.5) * env;
92   upper = SinOsc.ar(freq2, 0, 0.5) * env;
93   Out.ar(busLower, lower);
94   Out.ar(busUpper, upper)
95   }).add;
96
97
98 //this is not releasing properly
99 SynthDef(\accompTreble ++ ^hash, {arg freq, gate = 1, sustain, amp, bus;
100   var treble;
101   treble = SinOsc.ar(freq, 0, EnvGen.kr(Env.linen(0.3, 0, 0.7, amp * 0.075, \sine), gate, timeScale: sustain, doneAction: 2));
102   Out.ar(bus, treble)
103   }).add;
104
105
106 //----interlude
107 //note that this is sensitive to frequency and tremolo rate inputs
108 SynthDef(\interludeTremolo ++ ^hash, {arg gate = 0, amp = 1, freq1, freq2, tremRate;
109   var tremoloTrig, trem, freq, sig, feedback, fade;
110   //fast tremolo - note that this can be slower so long as the delaytime of the feedback remains short
111   tremoloTrig = Impulse.kr(tremRate);
112   //tremolo between two notes
113   trem = Select.kr(Stepper.kr(tremoloTrig, 0, 0, 1), [freq1, freq2]);
114   //occasionally tremolo on same note
115   freq = Select.kr(TWChoose.kr(Dust.kr(10), [0, 1, 2], [5, 1, 1], 1), [trem, freq1, freq2]);
116   //generate signal
117   sig = VarSaw.ar(freq, 0, 0.3, 0.1) * EnvGen.kr(Env.perc(0.01, 0.1), tremoloTrig);
118   //feedback
119   feedback = CombC.ar(sig, 0.2, tremRate.reciprocal, 5);
120   fade = feedback * EnvGen.kr(Env.asr(15, 1, 15, \sine), gate) * amp * 0.75;
121   Out.ar(\interludeTremoloBus, fade);
122   }).add;
123
124
125 //----gen music
126 genPatterns = {arg guitarSeqIn, accompLowSeqIn, accompHighSeqIn, sectionSeqIn, beatFrac = 1/8;
127   var calcSustains, genSectionSec, sectionLimits, measureCount;
128
129   //----helper sus function
130   calcSustains = {arg stringSeq, durSeq;
131     var res = [];
132     stringSeq.size.do({arg index;
133       var curString, dur, count;
134       if(stringSeq[index].isRest.not, {
135         curString = stringSeq[index];
136         dur = durSeq[index];
137         count = 1;
138         while(({stringSeq[(index + count).clip(0, stringSeq.size - 1)] != curString) &&
139           (dur < 16) && (count < 100)}, {
140           dur = dur + durSeq[(index + count).clip(0, durSeq.size - 1)];
141           count = count + 1;
142         });
143         res = res.add(dur.clip(0, 16));
144       }, {
145         res.add(Rest());
146       });
147     };
148   };
149   genSectionSec = {arg seq, startTime, endTime, type;
150     var durSum, resSeqs, inSecs, mult;
151     durSum = 0;
152     resSeqs = [];
153     seq.do({arg item;
154       if((durSum >= startTime) && (durSum < endTime), {
155         var dur = durSum - startTime;
156         if((resSeqs.size == 0) && (dur > 0), {
157           switch(type,
158             0, {resSeqs = resSeqs.add([Rest(-1), Rest(-1), dur])},
159             1, {resSeqs = resSeqs.add([Rest(-1), Rest(-1), dur])},
160             2, {resSeqs = resSeqs.add([Rest(-1), dur, dur])});
161         });
162         resSeqs = resSeqs.add(item);
163       });
164       durSum = durSum + if(type == 2, {item[1]}, {item[2]});
165     });
166     resSeqs
167   };
168
169   measureCount = 0;
170   sectionLimits = [];
171   sectionSeqIn.slice(nil, 0).add(100000).doAdjacentPairs({arg a, b; sectionLimits = sectionLimits.add([a, b])});
172   ^sectionStartMeasure = [];
173   sectionLimits.collect({arg timePair, secIndex;
174     var startTime, endTime, beatLength, beatSeq, measureSeq,
175     guitarSecSeq, accompLowSecSeq, accompHighSecSeq,
176     stringSeq, fretSeq, harmLimit, freqSeq, durSeq, susSeq, trigSeq, openStrings, pattern;
177
178     startTime = timePair[0];
179     endTime = timePair[1];
180
181     if((secIndex % 4) == 0, {measureCount = 0});
182     beatLength = (endTime - startTime) / 8;
183     beatSeq = ((beatLength / 2) - 1).asInteger.collect({[1, 2]});
184     beatSeq = if((beatLength % 2) == 0, {beatSeq.add([1, 2])}, {beatSeq.add([1, 2, 3])});

```



```

185 measureSeq = measureCount + beatSeq.collect({arg measure, mIndex; measure.collect({mIndex + 1})}).flat;
186 ^sectionStartMeasure = ^sectionStartMeasure.add(measureCount + 1);
187 measureCount = measureSeq.last;
188 beatSeq = beatSeq.flat;
189 measureSeq = measureSeq.add(0);
190 beatSeq = beatSeq.add(0);
191
192 guitarSecSeq = genSectionSec.value(guitarSeqIn, startTime, endTime, 0);
193 accompLowSecSeq = genSectionSec.value(accompLowSeqIn, startTime, endTime, 1);
194 accompHighSecSeq = genSectionSec.value(accompHighSeqIn, startTime, endTime, 2);
195
196 if (accompHighSecSeq == [], {accompHighSecSeq = [[Rest(-1), 1, 0], [Rest(-1), 1, 0]]});
197
198 openStrings = [1/1, 3/2, 2/1, 5/2, 35/12, 7/2];
199 harmLimit = [9, 8, 7, 6, 5, 4];
200 stringSeq = guitarSecSeq.slice(nil, 0);
201 fretSeq = guitarSecSeq.slice(nil, 1);
202 durSeq = guitarSecSeq.slice(nil, 2);
203 susSeq = calcSustains.value(stringSeq, durSeq);
204 freqSeq = stringSeq.collect({arg string, index;
205   if (string.isRest, {Rest()}, {
206     var midi, freq;
207     //this is transposed up because karplus-strong does not really sound correctly in the guitar range
208     midi = (62.midicps * openStrings[string]).cpsmidi + fretSeq[index];
209     freq = midi.midicps * if((secIndex % 4) != 3, {1}, {[1, harmLimit[string].rand + 1].choose});
210   });
211
212 pattern = EventPatternProxy.new;
213 pattern.source = Ppar([
214   Pbind(
215     \instrument, \karplus ++ ^hash,
216     \amp, 0.3,
217     \dur, Pseq(durSeq * beatFrac),
218     \sustain, Pseq(susSeq * beatFrac),
219     \freq, Pseq(freqSeq),
220     \bus, ^guitarBus.index),
221   if (accompLowSecSeq.size > 1, {
222     Pmono(
223       \accompBass ++ ^hash,
224       \amp, 0.5,
225       \freq1, Pseq(accompLowSecSeq.slice(nil, 0)),
226       \freq2, Pseq(accompLowSecSeq.slice(nil, 1)),
227       \dur, Pseq(accompLowSecSeq.slice(nil, 2)) * beatFrac,
228       \busLower, if(secIndex % 2 == 0, {^accompLowLowerBusA.index}, {^accompLowLowerBusB.index}),
229       \busUpper, if(secIndex % 2 == 0, {^accompLowUpperBusA.index}, {^accompLowUpperBusB.index}))
230     }, {
231       Pmono(
232         \accompBass ++ ^hash,
233         \amp, 0.5,
234         \freq1, Pseq([accompLowSecSeq[0][0]]),
235         \freq2, Pseq([accompLowSecSeq[0][1]]),
236         \dur, Pseq([accompLowSecSeq[0][2]]) * beatFrac,
237         \busLower, if(secIndex % 2 == 0, {^accompLowLowerBusA.index}, {^accompLowLowerBusB.index}),
238         \busUpper, if(secIndex % 2 == 0, {^accompLowUpperBusA.index}, {^accompLowUpperBusB.index}))
239     },
240   Pbind(
241     \instrument, \accompTreble ++ ^hash,
242     //\freq, Pseq(accompHighSecSeq.slice(nil, 0)),
243     \freq, Pseq(accompHighSecSeq.slice(nil, 0).curdle(0.3).collect({arg item; item.cpsmidi - 0.16 + 0.32.rand}).midicps.flat),
244     \dur, Pseq(accompHighSecSeq.slice(nil, 1) * beatFrac),
245     \sustain, Pseq(accompHighSecSeq.slice(nil, 2) * beatFrac),
246     \amp, 0.5,
247     \bus, ^accompHighBus.index),
248   Pbind(
249     \instrument, \transport ++ ^hash,
250     \measure, Pseq(measureSeq),
251     \beat, Pseq(beatSeq),
252     \dur, beatFrac * 8
253   )
254   );
255 pattern
256 ];
257
258 )
259
260 /*
261 //machine options
262 (
263   var durUnit = 0.15;
264
265   SynthDef(\machine, {arg freq, gate = 1, sustain, amp;
266     var sound;
267     sound = TWChoose.ar(Impulse.kr(0), [
268       PinkNoise.ar(EnvGen.kr(Env.perc(0.01, sustain, amp * 10), gate, doneAction: 2)),
269       BrownNoise.ar(EnvGen.kr(Env.perc(0.01, sustain, amp * 2), gate, doneAction: 2))
270     ], [0, 20], 1);
271     Out.ar([0, 1], sound)
272   }).add;
273
274   ^machine1 = Pbind(
275     \instrument, \machine,
276     \amp, Pseq(3000.collect({arg i; (i / 2000).clip(0, 0.03)})),
277     \dur, Pseq(1000.collect({[durUnit, durUnit + (durUnit / 100).rand2].wchoose([10, 1].normalizeSum)}.flat),
278     \sustain, Pseq(1000.collect({durUnit * (1.75 + 0.5.rand)}), //2.25].wchoose([1, 20].normalizeSum)})),
279     \freq, Pseq(1000.collect({[300, 250], [300, 250].choose].wchoose([10, 1].normalizeSum)}.flat / 2)
280   ).play;
281
282   SynthDef(\machine, {arg freq, gate = 1, sustain, amp = 0.03;
283     var trig, sound;
284     trig = Impulse.kr(10);
285     sound = BrownNoise.ar(EnvGen.kr(Env.perc(0.01, 0.3 + TRand.kr(0, 0.1, trig), amp), TDelay.kr(trig, TRand.kr(0, 0.002, Dust.kr(0.75)))));
286     Out.ar([0, 1], sound)
287   }).play;
288   )
289   */

```

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
111 musicData.do({arg part, p;
112   var amps, harm, modi, timeSigIndex, sectionCount, sectionIndex, subSectionIndex, curTimeSig,
113   lilyFile, lilyString, voices, lastVal, lilyNotes, lilyOcts, lilyGString, isHarmonic, measureCount,
114   lilyNote, lilyDur, lilyRest, lilyBeatingMark, curTime = 0, noteTuples, markupSuffixes;
115
116   //create file
117   lilyFile = switch(p,
118     0, {File(basePath + "/" + "includes" + "/" + "cicc.guitar.ly".standardizePath, "w")},
119     1, {File(basePath + "/" + "includes" + "/" + "cicc.high.ly".standardizePath, "w")},
120     2, {File(basePath + "/" + "includes" + "/" + "cicc.low.ly".standardizePath, "w")}
121   );
122
123   //start lilypond directives
124   lilyString = "";
125
126   lastVal = nil;
127
128   //start voice
129   lilyString = lilyString ++ "\n{ ";
130   lilyString = lilyString ++ "\n\\set Score.markFormatter = #format-mark-box-numbers ";
131
132   lilyString = lilyString ++ "\\tempo \\markup {\\concat {\\smaller \\general-align #Y #DOWN \\note #\"2\" #1 \\normal-text \" = approx. 90\"}}";

```

```
(p != 0) { lilyString = lilyString + "\override Staff.TimeSignature #'stencil?";  
lilyString = lilyString + "\\numericTimeSignature \\time 2/2\\n";  
  
lilyString = switch(p,  
    0, {lilyString + "\\clef \"treble(8)\"\\n"},  
    1, {lilyString + "\\clef \"treble(8)\"\\n"},  
    2, {lilyString + "\\clef \"bass(8)\"\\n"}  
);  
  
lilyNotes = ["c", "cis", "d", "dis", "e", "f", "fis", "g", "gis", "a", "ais", "b"];  
lilyOcts = [",", "", ",", "", ",", "", ",", "", ",,"], [{"", ""}, {"", ""}], [{"", ""}];  
  
timeSigIndex = 0;  
sectionCount = 0;  
sectionIndex = 1;  
subSectionIndex = 1;  
curTimeSig = 4;  
measureCount = 0;  
part.clump(4).do({arg beat, i;  
    var gSum = 0;  
  
beat.separate({arg a, b; ((a[0] != -1) || (b[0] != -1)) && (a != b)}).do({arg group, g; var noteLength, target = 0;  
noteLength = group.size;  
gSum = gSum + noteLength;  
  
//add ties  
lilyString = lilyString ++ if((p != 2) && (group[0] == lastVal) && (group[0][0] != -1), {"~ "}, {"});  
//add barcheck count  
lilyString = lilyString ++ if((curTime % curTimeSig == 0) && (i != 0), {measureCount = measureCount + 1; "| "}, {"});  
  
if(i == (sectionData[sectionCount][0] / 4) && (g == 0),{  
var barType, pageBreak;  
barType = switch(sectionData[sectionCount][1],  
    0, {"|"},  
    1, {"."},  
    -1, {"|"}.\\set Score.currentBarNumber = #1 "");  
pageBreak = switch(sectionData[sectionCount][1], 0, {""}, 1, {""}, -1, {measureCount = 0; "\\n\\pageBreak \\n \\time 2/2\\n"});  
isHarmonic = switch(sectionData[sectionCount][1], 0, {false}, 1, {true}, -1, {false});  
lilyString = lilyString + "\\bar " ++ barType ++  
" \\mark \\markup { \\bold \\box ++ sectionIndex ++ "." ++ subSectionIndex ++ } " ++ pageBreak;  
if(sectionCount < (sectionData.size - 1), {sectionCount = sectionCount + 1};  
switch(sectionData[sectionCount][1],  
    0, {subSectionIndex = subSectionIndex + 1},  
    1, {subSectionIndex = subSectionIndex + 1},  
    -1, {sectionIndex = sectionIndex + 1; subSectionIndex = 1})  
});  
  
if(i == (timeSigData[timeSigIndex][0] / 4) && (g == 0),{  
timeSigData[timeSigIndex][0];  
curTimeSig = timeSigData[timeSigIndex][1];  
if(curTimeSig % 2 == 0, {  
lilyString = lilyString + "\\n\\time " ++ (curTimeSig / 2).asInteger.asString ++ "/2\\n";  
}, {  
lilyString = lilyString + "\\n\\time " ++ curTimeSig.asString ++ "/" ++ 4 ++ "\\n";  
});  
if(timeSigIndex < (timeSigData.size - 1), {timeSigIndex = timeSigIndex + 1};  
curTime = 0;  
});  
  
switch(p,  
    0, {  
lilyNote = lilyNotes[((38.midiCps * openStrings[group[0][0]].cpsmidi + group[0][1]).round(1) % 12)];  
lilyNote = lilyNote + lilyOcts[((38.midiCps * openStrings[group[0][0]].cpsmidi + group[0][1]).round(1) / 12).asInteger - 2];  
},  
    1, {  
if(group[0][0] != -1, {  
lilyNote = lilyNotes[(group[0][0].cpsmidi).round(1) % 12];  
lilyNote = lilyNote + lilyOcts[(group[0][0].cpsmidi).round(1) / 12).asInteger - 2];  
}, {lilyNote = "x"});  
},  
    2, {  
if(group[0][0] != -1, {  
lilyNote = lilyNotes[(group[0][0][0].cpsmidi).round(1) % 12]; // * 2;  
lilyNote = lilyNote + lilyOcts[(group[0][0][0].cpsmidi).round(1) / 12).asInteger - 2];  
lilyBeatingMark = "" \\markup{ " ++ group[0][0][1].round(0.1) ++ " };  
}, {lilyNote = "x"});  
}  
});  
  
//duration  
lilyDur = switch(noteLength, 1, {"16"}, 2, {"8"}, 3, {"8.", 4, {"4"});  
//append rest directive  
//lilyRest = "";  
lilyGString = if((group[0] != lastVal) && (p == 0)), {  
var stringString, fretString;  
stringString = ["VI", "V", "IV", "III", "II", "I"][group[0][0]];   
fretString = group[0][1].asString;  
if(isHarmonic, {fretString = "\\musicglyph \"noteheads.s0harmonic\"});  
"\\.\\markup{\\concat{ " ++ stringString ++ " \\.\\super " ++ fretString ++ }} "  
}, {"");  
  
if((p != 2) || (lilyNote == "r"), {  
lilyString = lilyString ++ lilyNote ++ lilyDur ++ lilyGString;  
}, {  
lilyString = lilyString ++ "<<<{" ++ lilyNote ++ lilyDur ++  
" \\.\\laissezVibrer " ++ lilyBeatingMark ++ " }\\}\\}\\new Voice { \\voiceTwo " ++  
lilyNote ++ lilyDur ++ " \\.\\laissezVibrer }>>> \\oneVoice " ++ lilyGString;  
});  
  
//beam group  
if((p != 2) && (g == 0) && (noteLength != 4), {lilyString = lilyString ++ " [ "; });  
if((p != 2) && (gSum == 4) && (noteLength != 4), {lilyString = lilyString ++ " ] "); });  
  
lastVal = group[0];  
curTime = curTime + (noteLength / 4);  
});  
});  
  
//end voice  
lilyString = lilyString ++ " ] \\bar \".\\" } \\n";  
  
noteTuples = [lilyNotes, lilyOcts].allTuples.collect({arg val; val.join}).join("");  
  
markupSuffixes = ["VI", "V", "IV", "III", "II", "I"].collect({arg stringString;  
("\\\\\\\\\\\\\\\\musicglyph \\.\\noteheads.s0harmonic\\\\\\\\\\\\") ++ (0..14).collect({arg fret;  
"\\\\\\\\\\\\\\\\markup{\\concat{ " ++ stringString ++ " \\.\\super " ++ fret.asString ++ }}"))}).flatten.join("");  
  
lilyString.findRegexp(  
    "(" ++ noteTuples ++ ")^4 (" ++ markupSuffixes ++ ")" ^ "(" ++  
    "(" ++ noteTuples ++ ")^4 ^ "(" ++ noteTuples ++ ")^4 ^ "(" ++  
    noteTuples ++ ")^4 ^ "(" ++ noteTuples ++ ")^4 ^ "(" ++ noteTuples ++ ")^4"  

```

```

255     ).clump(6).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 r4").clump(2).do({arg match;
293       lilyString = lilyString.replace(match[0][1], "R1+3/2");});
294
295     lilyString.findRegexp("r4 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 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
308     lilyString.findRegexp("\\\\ r1").clump(2).do({arg match;
309       lilyString = lilyString.replace(match[0][1], "\\ R1");});
310
311     lilyString.findRegexp("4\\nrl").clump(2).do({arg match;
312       lilyString = lilyString.replace(match[0][1], "2\\n R1");});
313
314     //write file
315     lilyFile.write("{\\n" ++ lineBreakString ++ "}\\n" ++ lilyString);
316     lilyFile.close;
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"];
7    var trackNames = ["guitar", "high", "low 1", "low 2", "interlude"];
8    var partVols, partMutes, partPans;
9    var masterMute, masterVol;
10
11    // set initial mixer values
12    partVols = [1, 1, 1, 1, 1];
13    partMutes = [0, 1, 1, 1, 1];
14    partPans = [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".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       "play.set(\sel, currentSection % 2);
75       "patterns[sectionOrder[currentSection]].play("tempoClock, quant: 0);
76       if("interludes && ((currentSection % 4) == 3) && (currentSection != (sectionOrder.size - 1)), {
77         var center, interval, freq1, freq2, tremRate;
78         center = 50 - 12.0.rand;
79         interval = 3.0.rand + 2;
80         freq1 = (center + (interval / 2)).midicps;
81         freq2 = (center - (interval / 2)).midicps;
82         tremRate = 50 + 4.0.rand2;
83         "interludeTremelo.set(\gate, 1, \amp, 1, \freq1, freq1, \freq2, freq2, \tremRate, tremRate);
84       });
85       if((currentSection % 4) == 0, {
86         "interludeTremelo.set(\gate, 0);
87       });
88       if((currentSection % 4) != 0 && (currentSection % 4) != 3), {
89         "interludeTremelo.set(\gate, 0, \amp, 0);
90       });
91     }, {
92       var measure, beat;
93       measure = sectionStartMeasure[sectionOrder[currentSection]];
94       beat = 1;
95       updateTransport.value(measure, beat);
96     });
97   });
98 }.inEnvir;
99
100 // these funcs receive messages from the synth
101 OSCFunc({ arg msg, time;
102   {
103     var measure, beat;
104     measure = msg[3];
105     beat = msg[4];
106     updateTransport.value(measure, beat)
107   }.inEnvir.defer;
108 }, 'measureClock' ++ "hash, s.addr);
109
110 OSCFunc({ arg msg, time; {updateSection.value(1, false, false)}.inEnvir.defer}, 'nextSubsection' ++ "hash, s.addr);
111
112 OSCdef(\externalAdvance ++ "hash, {arg msg, time; {updateSection.value(1)}.inEnvir.defer}, 'nextSubsection', s.addr);
113
114 view = View(win);
115 generator = HLayout(
116   ranSeed = TextField(view, Rect(10, 10, 10, 20)).string("20200525"),
117   Button(view).states(["reset seed"]).action.({ ranSeed.string = "20200525"}.inEnvir),
118   Button(view).states(["random seed"]).action.({ ranSeed.string = 50000000.rand.asString}.inEnvir),
119   Button(view).states(["generate"]).action.({
120     "genAll.value(ranSeed.string.asInteger); "appStatus.string = "status: ready"}.fork(AppClock);
121     "appStatus.string = "status: generating".inEnvir,
122     Button(view).states(["transcribe"]).action.({
123       "transcribe.value(scoreData, ranSeed.string); "appStatus.string = "status: ready"}.fork(AppClock);
124       "appStatus.string = "status: transcribing".inEnvir,
125       ["appStatus = StaticText(view).string("status: ready"), stretch: 1], nil);
126   transport = HLayout(
127     Button(view).states(["<"]).action.({arg pState; updateSection.value(-4)}.inEnvir),
128     Button(view).states(["<"]).action.({arg pState; updateSection.value(-1)}.inEnvir),
129     Button(view).states(["play", "Color.black"], ["stop", "Color.black", "Color.grey"]).action.({
130       arg pState;
131       if(pState.value == 0, {
132         var measure, beat;
133         countOff.stop;
134         "isPlaying = false;
135         "patterns[sectionOrder[currentSection]].stop;
136         "interludeTremelo.set(\gate, 0);
137         measure = sectionStartMeasure[currentSection];
138         beat = 1;
139         updateTransport.value(measure, beat);
140         "interludeTremelo.set(\gate, 0, \amp, 0);
141       });
142       countOff = {
143         [1, 2, 1, 2].do({arg beat;
144           {
145             metronome.stringColor = metronomeColorFunc.value(beat);
146             metronome.string = metronomeStringFunc.value(beat);
147           }.defer;
148           0.75.wait;
149           {metronome.string = ""}.defer;
150           0.25.wait;
151         });
152         "isPlaying = true;
153         "play.set(\sel, currentSection % 2);
154         "patterns[sectionOrder[currentSection]].play("tempoClock, quant: 0);
155         if("interludes && (currentSection % 4) == 3) && (currentSection != (sectionOrder.size - 1)), {
156           var center, interval, freq1, freq2, tremRate;
157           center = 50 - 12.0.rand;

```

```

158         interval = 3.0.rand + 2;
159         freq1 = (center + (interval / 2)).midicps;
160         freq2 = (center - (interval / 2)).midicps;
161         tremRate = 50 + 4.0.rand2;
162         ~interludeTremelo.set(\gate, 1, \amp, 1, \freq1, freq1, \freq2, freq2, \tremRate, tremRate);
163     });
164     }.fork(~tempoClock, quant: 0);
165 }
166 }
167 }
168 Button(view).states([[>", Color.black]]).action({arg pState; updateSection.value(1)}.inEnvir),
169 Button(view).states([[>>", Color.black]]).action({arg pState; updateSection.value(4)}.inEnvir), nil,
170 sectionDisplay = StaticText(win).string("section: 1.1").font(Font("Monaco", 70)), nil);
171 view.layout(HLayout(
172     [VLayout(
173         HLayout(clock = StaticText(win).string(" 1.1").font(Font("Monaco", 200)),
174             StaticText(win).string("|").font(Font("Monaco", 200)),
175             metronome = StaticText(win).string.([[-30, -105, -104].asAscii].font(Font("Monaco", 300)).stringColor(Color.red)),
176             nil, transport, nil,
177             HLayout(
178                 tempo = TextField(view).string("90"),
179                 Button(view).states([["set tempo"]]).action({~tempoClock.tempo = tempo.string.asInteger / 60}.inEnvir),
180                 StaticText(view).string(" | "),
181                 Button(view).states([["auto advance", Color.black], ["auto advance", Color.black, Color.grey]]).action({
182                     arg v; ~autoAdvance = if(v.value == 0, {false}, {true}); ~autoAdvance;
183                 }).inEnvir).value(1),
184                 Button(view).states([["interludes", Color.black], ["interludes", Color.black, Color.grey]]).action({
185                     arg v; ~interludes = if(v.value == 0, {false}, {true})
186                 }).inEnvir),
187                 StaticText(view).string(" | "),
188                 address = TextField(view, Rect(10, 10, 10, 20)).string("127.0.0.1:57120"),
189                 Button(view).states([["set address:port"]]).action({
190                     var addr, ip, port;
191                     addr = address.string.split($:);
192                     ip = addr[0];
193                     port = addr[1].asInteger;
194                     thisProcess.openUDPPort(port);
195                     addr = NetAddr(ip, port);
196                     OSCDef(\externalAdvance ++ ~hash, {arg msg, time; {updateSection.value(1)}.inEnvir.defer}, '/nextSubsection', addr);
197                 }).inEnvir),
198                 [StaticText(view).string(" ") , stretch: 1]),
199                 [StaticText(view).string(" ") , stretch: 1],
200                 HLayout(
201                     order = TextField(view).string("1-16"),
202                     Button(view).states([["set order"]]).action({
203                         ~patterns["sectionOrder"]["currentSection"].stop;
204                         ~sectionOrder = order.string.split($,).collect({arg secEntry;
205                             var bounds;
206                             bounds = secEntry.split($-).collect({arg item; item.asInteger - 1});
207                             (bounds.minItem)..(bounds.maxItem).collect({arg sec;
208                                 (sec.asInteger * 4) + [0, 1, 2, 3]
209                             });
210                         }).flat;
211                         ~currentSection = 0;
212                         updateSection.value(0);
213                     }).inEnvir),
214                     [StaticText(view).string(" ") , stretch: 1]),
215                     [StaticText(view).string(" ") , stretch: 1], generator
216     ), alignment: \top));
217 faderView = {
218     var view, masterIndicators, trackIndicators, master, tracks;
219     view = View(win);
220     masterIndicators = {LevelIndicator()} ! 2;
221     trackIndicators = {LevelIndicator()} ! 5;
222
223     OSCFunc.new({arg msg; {
224         {arg i; masterIndicators[i].value = msg[3 + i].ampdb.linlin(-40, 0, 0, 1)} ! 2}.defer},
225         '/masterLevels' ++ ~hash, s.addr);
226     OSCFunc.new({arg msg; {
227         {arg i; trackIndicators[i].value = msg[3 + i].ampdb.linlin(-40, 0, 0, 1)} ! 5}.defer},
228         '/trackLevels' ++ ~hash, s.addr);
229
230     master = HLayout(
231         VLayout(
232             [HLayout(
233                 Slider(view).value(0.8).action({
234                     {arg v; masterVol = v.value * 1.25; ~play.set(\masterVol, masterVol)}.inEnvir),
235                     masterIndicators[0],
236                     masterIndicators[1]), stretch: 2],
237                 Button(view).states([["mute", Color.black], ["mute", Color.black, Color.grey]]).action({
238                     {arg v; masterMute = (1 - v.value).abs; ~play.set(\masterMute, masterMute)}.inEnvir),
239                 StaticText(view).string("master" .align(\center)
240             ), nil);
241     tracks = {arg part;
242         HLayout(
243             VLayout(
244                 HLayout(
245                     Slider(view).value(0.8).action({
246                         {arg v; partVols[part] = v.value * 1.25; ~play.set(partAbbr[part] ++ "Vol", partVols[part])}.inEnvir),
247                         trackIndicators[part]),
248                     Button(view).states([["mute", Color.black], ["mute", Color.black, Color.grey]]).action({
249                         {arg v; partMutes[part] = (1 - v.value).abs; ~play.set(partAbbr[part] ++ "Mute", partMutes[part])}.inEnvir).value({
250                             if(part == 0, {1}, {0}).value),
251                     StaticText(view).string("pan").align(\center),
252                     Knob(view).value(0.5).action({
253                         {arg v; partPans[part] = v.value * 2 - 1; ~play.set(partAbbr[part] ++ "Pan", partPans[part])}.inEnvir),
254                     StaticText(view).string(trackNames[part]).align(\center)
255                 ),
256                 nil)
257             } ! 5;
258         view.layout(HLayout(master, nil, *tracks));
259     helpView = {
260         StaticText(win).string(File.readAllString(~dir ++ "ciicc.readme.scd"));
261     };
262     tabButtonReset = {transportButton.value = 1; mixerButton.value = 1; helpButton.value = 1};
263     win.layout = VLayout(
264         HLayout(
265             HLayout(
266                 [
267                     transportButton = Button().states([["transport", Color.white, Color.grey], ["transport", Color.black]]).action({
268                         {tabButtonReset.value; transportButton.value = 0; tabs.index = 0 }.inEnvir).value(0), stretch: 1
269                 ], [
270                     mixerButton = Button().states([["mixer", Color.white, Color.grey], ["mixer", Color.black]]).action({
271                         {tabButtonReset.value; mixerButton.value = 0; tabs.index = 1 }.inEnvir).value(1), stretch: 1
272                 ]
273             ),
274             helpButton = Button().states([["help", Color.white, Color.grey], ["help", Color.black]]).action({
275                 {tabButtonReset.value; helpButton.value = 0; tabs.index = 2 }.inEnvir).value(1)
276         ),
277         tabs = StackLayout(masterView.value, faderView.value, helpView.value));
278     };
279 }

```

```

1 \version "2.19.83"
2
3 \include "cicc_pseudoincidents.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 = 2.5 \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 = 5
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-circled 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 "SpanBar.engraver"
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 ?

```