Making Echoes:

Utilizing the Markov Process in the Composition of Sound

BY

Eric Czekner

A thesis submitted in partial fulfillment of the
requirements for the degree of

MASTER OF ARTS IN
DIGITAL ART

at

GOUCHER COLLEGE
DECEMBER 2013

Adviser:
Christopher Willits

Readers:
Michael Scott-Nelson
Benjamin Taylor

Abstract

This work was designed as an experiment in the unique ways that Markov chains can be used in the composition of sound. During the construction of the project, two overall methods of using Markov chains were proposed; direct-composition and indirect-composition. Both methods were typically found to have satisfying results in terms of harmonic content, though some instances required influence from a human-composer. This further pushed the question of how much influence an artist should or should not have in the overall outcome of a work. Most of the hands-on manipulation of the outcome was the determination of which voices and instruments complemented one another musically. The research completed within the scope of this project provides solid examples in regards to the ability of utilizing Markov chains for sound composition. The findings of this experiment will prove useful for other musicians interested in unique compositional methods.

Overview Of *echoes*

Containing ten tracks, e*choes* is an album composed and created through the utilization of different variations on the Markov process. By using the process in a combination of direct and indirect ways, compositions were created to showcase the variety of methods in which the Markov process can be used for audio synthesis. The project combined both live instrumentation as well as computer programmed instrumentation to create a unique combination of sound compositions.

The overall intention of this experimental project was to immerse a listener into a world of ambient sound through the use of a mathematical process. However, it was not the intention for the listener to understand the methods utilized within the compositions. If a listener had a general appreciation for ambient and electronic music, the hope was to provide them with another album for their collection. If an individual has an understanding of the Markov process, this is an added bonus and will hopefully allow them to further appreciate the end result.

The intention of this thesis is to provide a thorough explanation of the methodology used in creating *echoes*. This will outline the techniques used in the actual content creation, including the explanation of the Markov process and its uses in creative realms, the use of Pure Data within the constraints of the project, and the methods used in creating the individual tracks.

The Markov Property and the Markov Process

Named for the Russian mathematician Andrei Markov, the Markov process is a process which meets the conditions of the Markov Property. Within a stochastic process, the Markov property refers to the memoryless property of the process (Markov, 1954). Each outcome is only dependent on the current value of the process, and not those preceding or succeeding the current value. In order for a stochastic process to be considered a Markov process, it must hold true to the Markov property.

The Markov process can be defined with slight differences depending on the order being referred to (first, second, etc.). However, as a general definition it can be accepted that a Markov process is, "a sequence of events in which the outcome at any stage depends on some probability" (Husen, 2013). In order for a process to be considered a Markov, three conditions must be met:

1. The number of possible outcomes or states is finite
2. The outcome at any stage depends only on the current state
3. The probabilities are constant over time (Husen, 2013)

Breaking this down into a mathematical equation can be seen like thusly: beginning with a state called *c*, which will be referred to as the *current state,* or the position of the process at a given moment (see rule 2 above). This state is crucial because the following is determined *only* by this current state (*c*). Next, there is a transition matrix which will contain the probabilities needed to determine how our process should move, which will be assigned value *M* (see rule 1 above). This transition matrix will always contain a finite number of possible outcomes. The process can seemingly go on forever, but it cannot be considered infinite because there will always be a discrete number of choices the system can make. A matrix may include millions of potential outcomes, but there will always be a finite number of potential results. Though the Markov process itself has the potential to loop infinitely, the process is finite given the fact that a discrete number of outcomes must exist for the process to be true.

Taking the two variables above, *c* for the current state and *M* for the transition matrix, the Markov process can be illustrated thusly:

$$c_{()}, Mc_{()}, M^2c_{()}$$

(fig. 1)

The current state (*c*) is the starting point. After one iteration the new state of the system becomes $Mc_{()}$, a second iteration results in $M^2c_{()}$ and so on. This has now become a Markov chain because the process continues to move through a number of possible outcomes whose probabilities are determined by the current state of the chain.

In order to satisfy Husen's third rule, the probabilities referred to within the process must remain constant over time. This means that as the process moves, the probabilities that were assigned to the potential outcomes must stay the same throughout the process.

$$c_1{:}c_2,\ \mathrm{P} = 30$$
$$c_2{:}c_1,\ \mathrm{P} = 10$$
$$c_1{:}c_3,\ \mathrm{P} = 20$$
$$c_3{:}c_1,\ \mathrm{P} = 05$$
$$c_2{:}c_3,\ \mathrm{P} = 20$$
$$c_3{:}c_2,\ \mathrm{P} = 15$$

(fig. 2)

This transitional matrix illustrates the current states, the outcomes, and the probabilities (P) as a percent. All the values of 'P' must remain unchanged while the process is moving in order for it to be classified as the Markov process.

Markov Chains, Their Probabilities and Results

A Markov chain is a specific utilization of the Markov process. Within a Markov chain, outcomes are provided with their given probabilities of occurrence. These probabilities can be user generated, set data, or a combination of both. In the above example (figure 2), the probabilities of outcome were user-generated. This means that they were created with no set data in mind, or in other words, simply made-up. While this provides a good example of exactly how a Markov chain functions, the outcomes generated won't always be very exciting, especially in terms of sound composition. This is not to say that the results will *always* be unexciting, because this may not be the case. The process of creating *echoes* has proven, however, that some of the most exciting results of a Markov chain will come from pre-determined sets of data.

For *echoes*, set data was used during the creation of all tracks. The reasoning for this was significant to the overall concept of the album. By utilizing pre-existing sets of data to create Markov chains, the album essentially creates echoes of the source material. While the original material is often indistiguishable from the new content, the fact that this process was chosen was crucial to the overall production of the project. Data sources used for the creation of the album were composed from three categories:

- Existing MIDI files of popular classic rock and pop-punk songs
- A collection of one month of power ball drawing numbers
- A collection of Facebook friend's alphabetically arranged birthdays

The most successful result achieved was done so through the use of the pre-existing MIDI files. This provides the reasoning as to why seven of the ten tracks utilized this methodology for composition. Furthermore, it helps prove the idea that the use of pre-existing data sets can make for more harmonic results when Markov chains are used in sound composition.

Graphical Representations of Markov Chain Data

Figure 2 displays one way of graphically representing a Markov chain. To examine this graph more in depth, it can be broken down line-by-line. However, each line provides the same essential structure so it is not necessary to explain every line. Line one shows our current state (c_1) moving to the second state (c_2) with a probability of 30%. These placeholders can be assigned numbers to better illustrate how the system works. Looking at sound composition through MIDI, (c_1) could be assigned MIDI note 60, while (c_2) could be assigned MIDI note 61. This means, that anytime MIDI note 60 is played, the probability that the next note will be MIDI note 61 is (and always will be within the confines of the chain) 30%.

This method of graphically visualizing Markov chains is usable, but there are a couple of other methods which can be used. Many other methods provide a clearer way of presenting the data to a viewer in order to give a better understanding of the included probabilities. Before doing that, it will be helpful to assign each value in the system a MIDI note for clarity.

$C_1 = 60$
$C_2 = 61$
$C_3 = 62$
(fig. 3)

Taking the data from figure 2, these numbers can be translated into two different methods for displaying data of a Markov Chain. Below, figure 4 shows one of the ways this could be represented.

	0	**60**	**61**	**62**
60	50	0	30	20
61	70	10	0	20
62	80	5	15	0

(fig. 4)

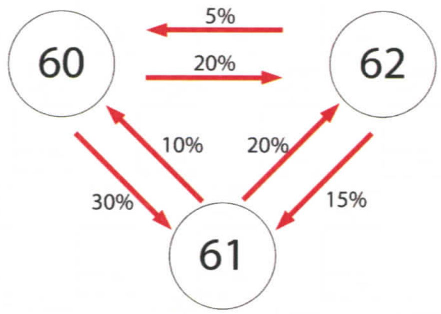
This type of graphical representation is known as a stochastic matrix, a transition matrix, or a probability matrix. These terms should be considered interchangeable. In this graph, the inputs (or current states) are displayed in the left columns, the results are displayed in the top row and the outcomes are displayed in middle and right columns. It is important to note that this represents a non-looping chain because there exists a certain probability that the chain will terminate itself. In this case, the likelihood that the chain will terminate is rather high. If the chain needed to be looping, the probability alignment may look like this:

	60	**61**	**62**
60	50	30	20
61	10	70	20
62	5	15	80

(fig. 5)

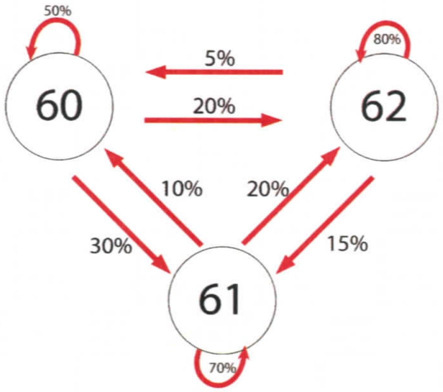
In figure 5, each line is equal to 100 %, which means that the chain will not terminate itself at any time. There will always be a place for the note in the current state to move to.

Another way to graphically represent a Markov chain is through a directed graph. The example below uses the same data-set from before.



(fig. 6)

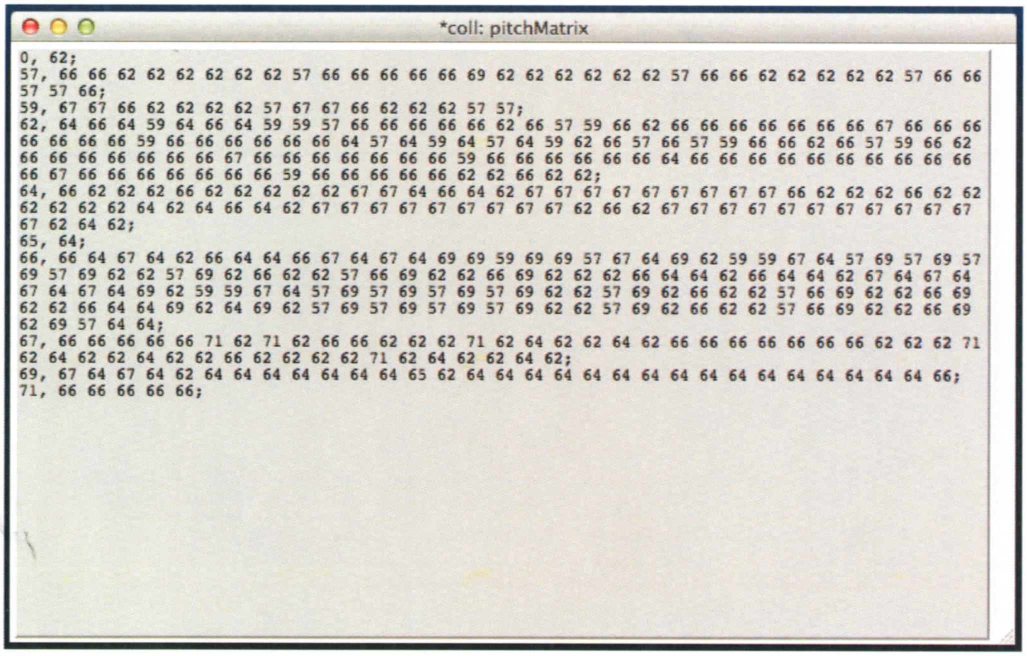
As in figure 4, this chain will terminate itself at a certain point. Using the directed graph, it is possible to represent the looping Markov chain in the following way.



(fig. 7)

All of the previous examples show the same Markov chain in different forms, with the exceptions of figures 5 and 7, which are altered to be looping Markov chains that never terminate. The method used to graphically represent a Markov chain is entirely up to the person using it, and these examples are not the only methods in which a chain can be shown. Once the sets of data begin reaching hundreds of numbers, it can become tedious to view graphs in this way.

Through the creation of *echoes,* the matrix display used was controlled by constructed Pure Data patches. It closely resembles the look of the Stochastic matrix in figures 4 and 5, with a slight difference; probabilities are displayed as strings of outcomes rather than percentages. It can become confusing to look at once the data-sets are large, but below is a sampling of one of the matrices used.



```
0, 62;
57, 66 66 62 62 62 62 62 62 57 66 66 66 66 66 69 62 62 62 62 62 62 57 66 66 62 62 62 62 62 57 66 66
57 57 66;
59, 67 67 66 62 62 62 62 57 67 67 66 62 62 62 57 57;
62, 64 66 64 59 64 66 64 59 59 57 66 66 66 66 66 62 66 57 59 66 62 66 66 66 66 66 66 66 67 66 66 66
66 66 66 66 59 66 66 66 66 66 66 64 57 64 59 64 57 64 59 62 66 57 66 57 59 66 66 62 66 57 59 66 62
66 66 66 66 66 66 66 67 66 66 66 66 66 66 66 59 66 66 66 66 66 66 64 66 66 66 66 66 66 66 66 66 66
66 67 66 66 66 66 66 66 66 59 66 66 66 66 66 62 62 66 62 62;
64, 66 62 62 62 66 62 62 62 62 62 67 67 64 66 64 62 67 67 67 67 67 67 67 67 67 66 62 62 62 66 62 62
62 62 62 62 64 62 64 66 64 62 67 67 67 67 67 67 67 67 67 62 66 62 67 67 67 67 67 67 67 67 67 67 67
67 62 64 62;
65, 64;
66, 66 64 67 64 62 66 64 64 66 67 64 67 64 69 69 59 69 69 57 67 64 69 62 59 59 67 64 57 69 57 69 57
69 57 69 62 62 57 69 62 66 62 62 57 66 69 62 62 66 69 62 62 62 66 64 64 62 66 64 64 62 67 64 67 64
67 64 67 64 69 62 59 59 67 64 57 69 57 69 57 69 57 69 62 62 57 69 62 66 62 62 57 66 69 62 62 66 69
62 62 66 64 64 69 62 64 69 62 57 69 57 69 57 69 57 69 62 62 57 69 62 66 62 62 57 66 69 62 62 66 69
62 69 57 64 64;
67, 66 66 66 66 66 71 62 71 62 66 66 62 62 62 71 62 64 62 62 64 62 66 66 66 66 66 66 66 62 62 62 71
62 64 62 62 64 62 62 66 62 62 62 62 71 62 64 62 62 64 62;
69, 67 64 67 64 62 64 64 64 64 64 64 64 65 62 64 64 64 64 64 64 64 64 64 64 64 64 64 64 64 64 66;
71, 66 66 66 66 66;
```

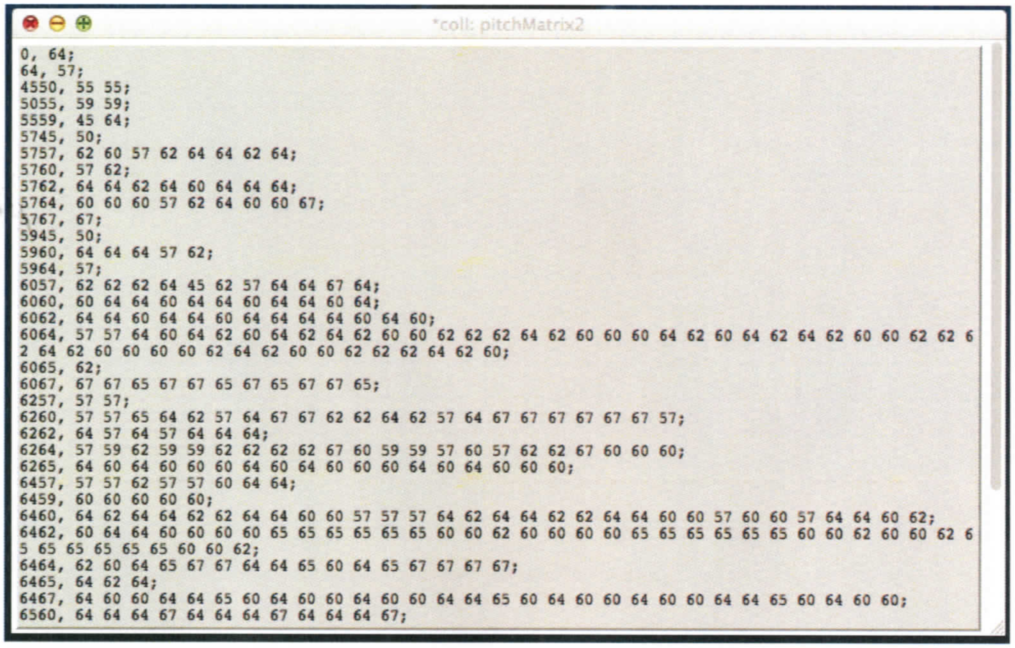
(fig. 8)

In figure 8, the current states are listed first, followed by a comma. After the comma are all of the possible outcomes for the chain to move to. Each line is terminated by a semi-colon. The first line displays that the current state of 0 (MIDI note 0), has a 100% probability of moving to MIDI note 62. Further down, MIDI note 62 shows its list of potential outcomes. Pure Data is able to recognize this type of format for its Markov matrices and this was used exclusively in the creation of *echoes.*

Understanding First and Second Orders

The examples of Markov chains this far have been exclusively considered first-order chains. This means that the chain only takes into account the current state it is in to determine the next movement it should make. This holds true to a condition required for the Markov property to exist; the system must be memoryless (Markov, 1954). Another way of looking at this, is that "...what happens in the long run won't depend on where the process started or on what happened along the way. What happens in the long run will be completely determined by the transition probabilities – the likelihoods of moving between the various states" (Page, 2012).

Another layer of the Markov process that exists and bends this rule is called second order chains. When referring to a Markov chain, the term "order" applies to the number of past events the chain will use to make its determination of where to go next (Taube, 2004). In a second order Markov chain, the next state is determined by the previous *two* states that the chain was in. This does somewhat disprove the fact that a Markov chain must be memoryless, because a certain level of memory is needed to remember the current state along with the most previous state before it. However, once the number is moved out of the second order level, it is then essentially forgotten by the system, returning the process to the requirement of being memoryless. Figure 9 illustrates the Pure Data matrix of a second-order Markov chain.



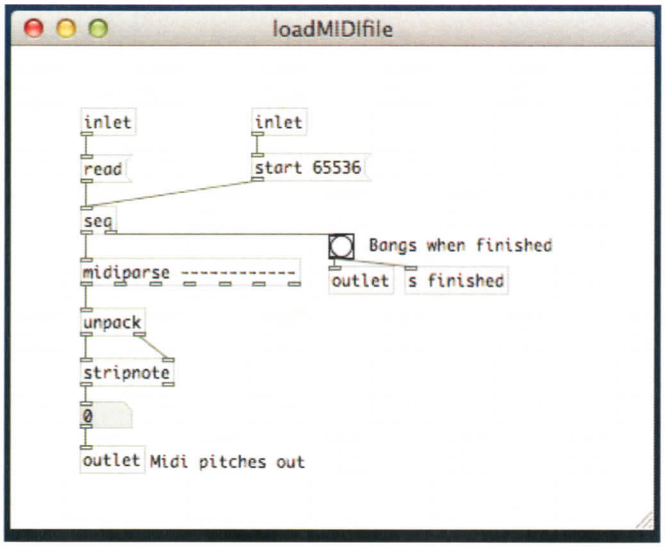
```
0, 64;
64, 57;
4550, 55 55;
5055, 59 59;
5559, 45 64;
5745, 50;
5757, 62 60 57 62 64 64 62 64;
5760, 57 62;
5762, 64 64 62 64 60 64 64 64;
5764, 60 60 60 57 62 64 60 60 67;
5767, 67;
5945, 50;
5960, 64 64 64 57 62;
5964, 57;
6057, 62 62 62 64 45 62 57 64 64 67 64;
6060, 60 64 64 60 64 64 60 64 64 60 64;
6062, 64 64 60 64 64 60 64 64 64 64 60 64 60;
6064, 57 57 64 60 64 62 60 64 62 64 62 60 60 62 62 62 64 62 60 60 60 64 62 60 64 62 64 62 60 60 62 62 6
2 64 62 60 60 60 60 62 64 62 60 60 62 62 62 64 62 60;
6065, 62;
6067, 67 67 65 67 67 65 67 65 67 67 65;
6257, 57 57;
6260, 57 57 65 64 62 57 64 67 67 62 62 64 62 57 64 67 67 67 67 67 67 57;
6262, 64 57 64 57 64 64 64;
6264, 57 59 62 59 59 62 62 62 62 67 60 59 59 57 60 57 62 62 67 60 60 60;
6265, 64 60 64 60 60 60 64 60 64 60 60 60 64 60 64 60 60 60;
6457, 57 57 62 57 57 60 64 64;
6459, 60 60 60 60 60;
6460, 64 62 64 64 62 62 64 64 60 60 57 57 57 64 62 64 64 62 62 64 64 60 60 57 60 60 57 64 64 60 62;
6462, 60 64 64 60 60 60 60 65 65 65 65 65 65 60 60 62 60 60 60 60 65 65 65 65 65 65 60 60 62 60 60 62 6
5 65 65 65 65 65 60 60 62;
6464, 62 60 64 65 67 67 64 64 65 60 64 65 67 67 67 67;
6465, 64 62 64;
6467, 64 60 60 64 64 65 60 64 60 60 64 60 60 64 64 65 60 64 60 60 64 60 60 64 64 65 60 64 60 60;
6560, 64 64 64 67 64 64 64 67 64 64 64 67;
```

(fig. 9)

In this matrix, rather than a single MIDI note in the current state position, there are two notes. In line three, the number 4550 is shown, again followed by a comma. Whenever the sequence of notes is 45 followed by 50, the next note will be 55. Again the lines begin with the states (in this case the current and the one that directly preceded it) followed by a comma, another set of notes and then a semi-colon to terminate the line. This illustrates the difference between how a first and second order chain functions. In a second order chain, the system is still constrained by a finite number of outcomes, and the probabilities must always remain unchanged through the duration of the process. Otherwise, this could no longer be considered a true Markov process.

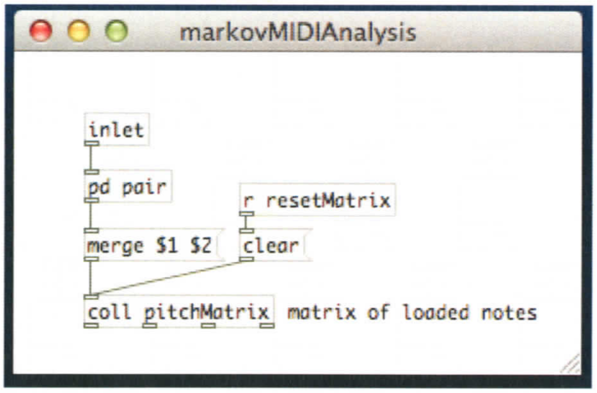
The Pure Data Patch

The patch itself consists of two parts: a player and a matrix loader. The core of each was compiled with the help of some useful articles found at algorithmiccomposer.com. The matrix loader is an abstraction which appears inside the player and is used before anything of value can happen within the patch. Without loading data into a matrix, there is no way for the system to create the necessary chains. First, the user must open a MIDI file, which is then loaded by the 'loadMIDIfile' sub-patch illustrated below in figure 10.



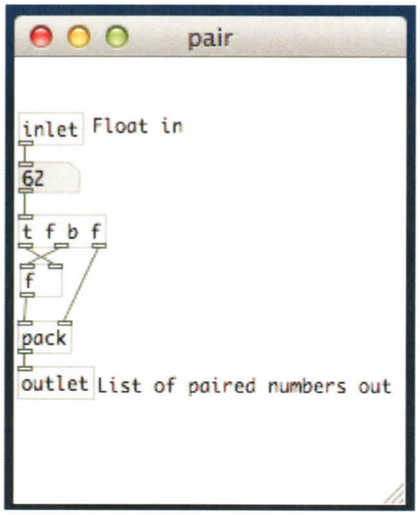
(fig. 10)

This sub-patch cues the MIDI file and waits for a signal to be sent from the "loads into coll matrix" command in the matrix loader abstraction. Once the signal is sent, the MIDI data is fed through the 'markovMIDIanalysis' sub-patch shown in figure 11 below. This sub-patch sends the MIDI file sequentially through another sub-patch called 'pair'. Figure 12 illustrates the contents of the pair sub-patch where the actual calculation of the data takes place. Within 'pair' the numbers are read through as a sequence, and each number is paired with the number following it via the [pack] object. These numbers are then sent through the outlet and into the merge method where they are placed inside of the [coll pitchMatrix] object (fig. 9). The coll matrix is the transition matrix where all of the MIDI streams will be generated from.



```
inlet
pd pair
                r resetMatrix
merge $1 $2     clear
coll pitchMatrix  matrix of loaded notes
```

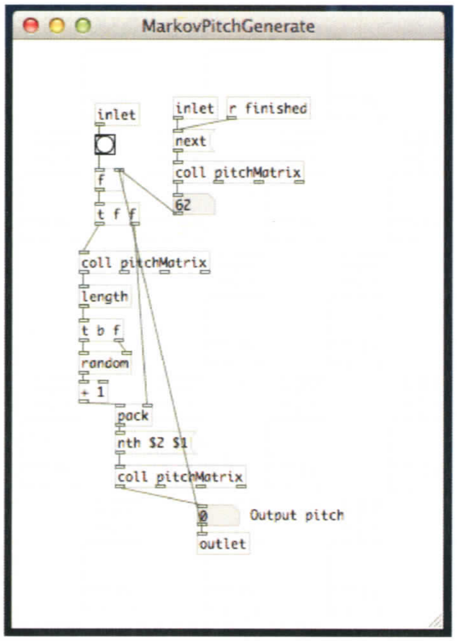
(fig.11)



```
inlet Float in
62
t f b f
f
pack
outlet List of paired numbers out
```

(fig.12)

The final section of the patch is the MarkovPitchGenerate sub-patch shown in figure 13. This is the area that will allow for the playback of the information from the transition matrix. It functions by looking at the number that is in the current state, finding it within the matrix, and then choosing a number to move to from the probabilities stored in the matrix. The number that was selected is then loaded into the top most [float] object to be used as the current state.



(fig. 13)

Composing *echoes*: A "Track-by-Track" Breakdown

Each track on the album used similar techniques in the form of the software used. Pure Data was incorporated into every track, but not always in the same method each time. In some pieces Pure Data is used to compose the entire song by outputting MIDI notes into Ableton Live; a process now referred to as 'direct composition'. On other tracks live guitar samples were recorded while Pure Data output MIDI data from Markov chains to control the parameters of musical effects. Finally, a third method used a live-performance setup with Pure Data displaying numbers corresponding to effects on guitar tracks. These effects were toggled on or off when the matching number was output from the patch. The latter two processes are being labeled as 'indirect composition'. This mixture of techniques allowed for increased exploration of the limits and abilities of composing with Markov chains.

Ableton Live was used as the main composition tool for every track on the album. The ease of sending MIDI from Pure Data to Ableton made this a necessity. Ableton also provided for a large range of effects, pre-built instruments and a real-time method for recording the incoming information from Pure Data patches.

Many of the tracks on *echoes* use very similar methodology in their creation. Therefore, the following provides an in-depth breakdown of only some tracks in order to illustrate the main differences and discoveries made during the creative process. Each track will be described, but some explanations will be less involved than others because of the similarities in the ways the tracks were created. However, across the album, the MIDI files and instrumentation used is altered, allowing for the same technique to generate a wide array of compositional outcomes.

- Track 1: For The Stories

For The Stories began with a MIDI file from the group Fall Out Boy's song titled "Champagne For My Real Friends, Real Pain For My Sham Friends." The original MIDI file included six instrument voices; two guitar tracks, a bass track, drums, lead vocals, and backing vocals at a tempo of 170 bpm. In order to focus on a more concise set of data for the Markov chains, only one of these MIDI tracks was used. In this case, the lead vocal track was utilized. This proved to be useful to keep the melody somewhat contained. It was found that using rhythm guitar tracks sometimes led to unnatural or non-melodic sounding output. Therefore, it was a good idea to keep it somewhat simple and focus on using one or two 'voices' from the original file.

The lead vocal MIDI track was output from Ableton and run through the first-order Markov generator patch in Pure Data. Next, a new MIDI 'voice' was set up in Ableton and the Pure Data patch was started. This sent a stream of MIDI notes to the new voice at a tempo of one note every 705.8 milliseconds, equivalent to a half note at the 170 bpm rate of the original song. The length of each MIDI note was 1411.6 milliseconds, or equal to that of a whole note at 170 bpm. This was defined in the Pure Data patch. There was an additional control added to this voice in the way of an effect parameter modulation. The 'depth' parameter on the 'oct+six mod' effect was automated to a random number between 0 and 50. Each time the Markov chain chose the number 60, random numbers between 0 and 50 would be output from Pure Data to control the effect parameter. This occurred at a rate of 176.45 milliseconds, equivalent to an eighth note at the given tempo. The reasoning for this was to provide an increased chance of change during the playback of the Markov chain. This was found to increase the overall depth of the track by adding an additional variable that could be changed.

With one track established, an additional voice was needed to add further variety. Therefore, another instrument was added, and the same MIDI vocal file was loaded into Pure Data. This time, it was processed through the second-order patch to add further harmonic depth. In the same fashion as the first track, these numbers were output at a rate of one every 705.8 milliseconds, with a length of 1411.6 milliseconds. Outputting the stream to a separate MIDI channel (2) allowed this new set of numbers to run simultaneously with the first set while not interfering with the data. This was crucial to adding depth and dimension to the piece. By creating an entirely different voice within the track, it provides a secondary level of variety in the harmonies.

The piece itself is meant to be a relaxing, ambient track that builds slightly throughout while providing the listener with a subtle rhythmic panning. The panning effect almost hints along the lines of a drum track, even though none exists within the work. The title of the song, as with many included on the album, is a reference to a section of lyrics in the original piece of music.

-Track 2: Mountains Over Malibu

This piece uses a Markov chain created from an alphabetized list of Facebook friends' birthdays. To do this, friends were arranged alphabetically and the transition of months was collected. For example, if person 'A' had a birthday on May 14th (5/14), and person 'AB' had a birthday on February 10th, (2/10) the result 5, 2 was added to the transition Matrix. While this task was tedious, and one that eventually could be automated, it provided a solid set of data to work with. However, because the data did not start as a musical element itself, it was not used strictly in a musical sense of output. Rather, the numbers were used to toggle on and off MIDI control changes.

In the Pure Data patch, once the transition matrix is loaded, it outputs a stream of numbers. When the number 5 is selected, a [metro] object is toggled on and random numbers from 0 to 100 are output at a rate of one every 100 milliseconds. These numbers send MIDI CC messages to track four's delay effect, specifically the feedback parameter. When the number 5 is selected from the matrix a second time, the [metro] object is toggled off and the stream of notes is stopped. Similarly, when the number 12 is selected, a [metro] object at 1000 milliseconds is toggled on. This starts an output of a stream of random numbers between 0 and 50 to track 1one's oct+six mod effect's formant parameter. Once again, the next time 12 is selected this is toggled off.

The music itself was done as a study of simplicity in composition. To do this, a single MIDI note was used. MIDI note C3 was played back by four different instrument tracks, each at a length of 2 whole notes (at 110 bpm). All four tracks were set as different instruments and the effects applied to each are different. A fifth track is included and used for the intro and conclusion of the piece. It was found that when the Pure Data patch was running, track number four would occasionally have an output even if the instrument was not playing back. This 'extra noise' was routed to a fifth instrument which included effects of its own and used to create an additional voice within the piece. Finally, a sequenced drum track was added for the inclusion of a beat through some of the song.

One major setback to using this process was that the matrix needed to be loaded each time the patch was opened. In further exploration, this may be able to be solved, but proved to be only a small nuisance while completing the overall piece.

- *Track 3: Driving Towards Orion*

Like *Mountains Over Malibu*, this piece was created using the indirect composition technique. However, it added the variable of a live performance mechanism in the form of a visual patch using GEM in Pure Data. This song used the same list of birthdays as the track before it, but rather than the Markov chains influencing the effect parameters, they were used to indicate to a performer when to turn on and off certain effects.

The Ableton setup uses a total of six guitar tracks at a tempo of 100 bpm. There are three input guitar tracks and three output guitar tracks. Mapping the pedals on a Behringer FCB1010 to toggle effects, a musician can use the patch in conjunction with the Live setup to create an ever changing performance environment. The numbers in the Markov chain each refer to an effect in a chain on the guitar tracks. When the number corresponding to a given effect appears in the GEM window, that effect should be toggled on or off depending on its previous position. The effects can be rearranged or edited by the performer depending on the desired outcome.

- *Track 4: Aim All The Spotlights*

This track uses a single MIDI file again from the band Fall Out Boy. The original piece, "A Little Less Sixteen Candles, A Little More Touch Me" contained two guitar tracks, a drum track and a vocal track. *Aim All The Spotlights* uses only the vocal track from the original source, and uses only a single new instrument in the final outcome.

Although the expected result of using a single voice to create this piece was not necessarily a musical one, this proved to be incorrect when a minor change was made in the Pure Data patch. Rather than having the MIDI notes output at a steady stream (usually equal to the tempo of the original piece), this track added a method for the Markov chain to change the pattern of the output notes. By using the [select] object along with three notes occurring in the chain, the tempo is able to be changed whenever a certain note is in the current state. The inclusion of this change allowed for an increase in the melodic complexity of the song and proved to be crucial to successes of other tracks on the album. Adding a further variety to the rhythms within the stream of MIDI notes allowed for greater depth in the final sound of a piece.

A final addition to this piece was the ability for the Markov chains to control the length of the notes being output. Again, by using the [select] object, each time a certain number was in the current state, the note length would be changed to a random number between 450 and 5000 milliseconds. In combination with the changing of the tempo throughout the piece, this helped to add an additional method of creating unique rhythmic outcomes.

- *Track 5: We Danced, We Laughed, We Love*

Track number five used direct composition techniques similar to track one. However, rather than one MIDI file controlling two tracks, three MIDI files each control their own instrument. Using a MIDI file from Journey's "Don't Stop Believin'", the Pure Data patch was altered to contain three separate transition matrices. Next, a MIDI file containing the data from the original piano track was loaded into its own matrix. Finally, MIDI files from both guitar tracks were each run through their own matrices. Again, by routing each Markov chain to its own instrument, a variety of harmonies were developed.

Also present in this track is the indirect use of the chains similar to those in songs two and four. Like in track two, each Markov chain was responsible for toggling on and off [metro] objects which would change certain effect parameters of the tracks. Similar to song four, this piece also used indirect composition with the chains to periodically alter the tempos of the streams coming from the guitar track Markov chains.

- *Track 6: You've Built God, Now Let's Talk*

Similar to the previous tracks, track six used direct composition techniques in combination with indirect composition. In this piece the main melody was created by second-order Markov chains from a MIDI file of Panic! At The Disco's "I Write Sins, Not Tragedies". Within the original piece, a music-box is present. This is the specific MIDI track that was used to control the melodies in *You've Built God, Now Let's Talk.* The same MIDI track controls the secondary melody through the use of a first-order chain. This led to a couple of important discoveries being made during the creation of this piece.

It was found that the original music-box MIDI file had very little to offer in terms of melodic depth. The note structure is fairly limited and simplistic. However, once this was run through a second-order chain coupled with the indirect composition method of controlling the tempos, it appeared to be the opposite. This led to the belief that second-order chains seemed to make for better melodic composition. Additionally, the smaller the data-set the chain was created from, the more capable of creating melodic-depth it seemed to have.

- Track 7: Together We Get Older

The Pure Data patch for this piece is much more involved than the previous tracks. This patch is designed to work with a LPD8 midi controller pad in a live-performance environment. The functionality of the patch is the same as every other one used, except the on/off events of the Markov chains can be controlled using the LPD8 controller. *Together We Get Older* uses three tracks from a MIDI file from The Who's "Baba O'Riley" to create the Markov chains; all of which happen to be synthesizer tracks. These MIDI streams are then output as first-order chains into different instrument voices within Ableton.

The techniques used here were very similar to the other pieces on the album, with a slight difference. In order to create the somewhat ominous introduction of this track, the MIDI stream of the first synthesizer file was output with a [metro] object set at a rate of 1 millisecond. This was unintentional at first, but proved to provide a unique sounding result; one that the piece definitely benefited from.

-Track 8: Rethinking Jesus

Track eight is one of the three pieces on the album which utilize only live instrumentation in the composition. It is also one of the tracks which had the least amount of influence from the Markov process. The intro guitar riff is from the band Brand New's song, "Jesus Christ". The intention of this piece was to begin with a recognizable guitar riff and allow it to fade away using heavy effect processing. The only place that Markov chains are used is to control the 'amount' parameter on the frequency shifter effect of 'guitar 2 loop'. While the impact is subtle, the track is still successful at its intention. This helps prove that even with little influence, Markov chains can be useful in this method.

-Track 9: Tomorrow When The Sun Comes Up

The second-to-last track was another pursuit of simplicity within sound composition. Using a single instrument and direct composition from a first-order Markov chain, the track provides for a relaxing ambient music bed. The piece used the main guitar track from a MIDI file of Blink 182's

"Adam's Song". The new stream of MIDI notes were output at a steady whole note rate. The combination of two delay effects on the new instrument helped maintain the drawn-out ambient sound of the piece.

-Track 10: The Skyline Will Be Beautiful On Fire

The final song on *echoes* is another using only live instrumentation, with the exception of the MIDI drum track added for rhythmic variety. Markov chains are again only being used in an indirect fashion, much like track eight. The chains control only the pitch decay parameter on the beat repeat effect of guitar 5.

The matrix for this was created using a month's long collection of Power ball drawing numbers. The intention behind this was to look for a data-set that existed in the real world and was not geared towards a musical output. Any numbers that contained two digits (i.e., 35) were separated into their single digit numbers (3, 5) to add a larger variety to the set. Once again, the use of this type of matrix proved to be better when applied to an indirect-composition method. Rather than the numbers of the chain controlling the MIDI notes and melody, it was much better to allow them to control other parameters of the composition.

Findings and Observations

A few important findings were made during the creation of *echoes*. Possibly the most important fact being that Markov chains can be used in both direct and in-direct methods of composition, with equally satisfying results. The combination of both methods can prove to have an even greater influence on the outcome of a piece. However, the utilization of these methods led to a lot of questioning in regards to the amount of control an artist should have over his or her work.

When working with the direct method in the way it was used for this album, the artist's control is limited to a few things. First, they have the decision of the MIDI track from which the data-sets will be pulled. Second, they can decide on which, and how many new instrument voices should appear. Finally, the artist has the control of the effects used the further process each instrument track. Once the MIDI files are loaded into Pure Data and playback is started, the system determines where it should go. While the artist has the option to start or stop the system, or pull certain 'riffs' that sound best, the direct method of composition with Markov chains leaves very little control in the hands of the artist. This is not necessarily a bad thing, and as *echoes* has proven, it can lead to some interesting results. However, some may feel that artists should have more control over the work they are creating.

In this form of creation, the artist is somewhat limited by the information that is initially fed into the system. If the MIDI files that are chosen for the Markov chains lack strong melodic-depth, the system is often unsuccessful in creating a truly harmonic result. This was found to be the case with a lot of guitar tracks from pop-punk songs. While some did work well, others resulted in monotonous sounding patterns which lacked a certain depth.

When using an in-direct method of composition, especially coupling it with live instrumentation, the control is much more in the hands of the artist. By allowing the artist to compose an entire piece that could essentially stand on its own and using Markov chains only to influence certain effect parameters, the artist had far more influence on the outcomes. "Rethinking Jesus" and "The Skyline Will Be Beautiful On Fire" are fitting examples of the fact that the combination of indirect influence from Markov chains and live instrumentation can generate beautiful results.

A second discovery that was made during the creation of *echoes* was the fact that combining a smaller data-set with direct composition through second-order chains provided for the most melodic outcomes, as the track "You've Built God, Now Let's Talk shows. The MIDI data from the music-box track of the original file had very little in terms of numerical possibilities. When this was processed through a second-order chain and sub-sequentially fed through a similar sounding instrument, the result was highly melodic.

While the main reasoning for this is uncertain, it may have something to with the fact that melodies are simply patterns of a set number of notes in a constant order. When the data-set used is smaller, it is much more likely that the same result will occur within the chain, creating a repetitive pattern similar to a common melody within a song.

The final important discovery made during this process was that the inclusion of a way for the patterns' tempos to be altered allowed for more complexity in the results. This idea was taken further with the addition of the ability for the note lengths to be changed. Rather than having a steady stream of notes at a given tempo and of the same length, the variations created through these techniques enhanced the melodic structures of the songs.

The Future

Initially, two performances of the album had been planned to culminate the creative experience. The methods used in the process did not lend themselves well to live-performances. While some tracks are set up to incorporate performance elements, the future looks promising that this will eventually be achieved. Within the construct of this specific album, however, there were not enough live-performance set ups created to warrant a live showing. The challenge of using Markov chains in performances that truly engage the audience has not yet been conquered and was the ultimate cause for the performances to be removed from the project.

The project also initially called for a video element to be included with each completed track. These videos were intended to be controlled by the Markov process, and to match up with the audio of each pieces. Once the recording process of the album was started this proved to be an overwhelming undertaking with respect to the work already being done with the audio. The ultimate goal is to somehow use Markov chains in both audio and video creation, which could then be translated to a live performance environment.

The process involved in creating this album was arduous, though the end result was incredibly rewarding. Even though some concepts that were proposed at the beginning of the project were not included, the album still proves the diversity of the Markov process in sound composition. Many areas surrounding the ways which Markov chains can be used are yet to be discovered. Despite this, *echoes* proves that the utilization of the Markov process in sound composition is a viable method for content creation. The possibilities of using Markov chains for composition, live performance and video creation are seemingly endless, and e*choes,* while an endeavor in itself, barely scratches the surface of the work that is yet to come.

References

Husen, B. (2013). *Markov processes*. Retrieved from https://people.math.osu.edu/husen.1/teaching/571/markov_1.pdf

Markov, A. A. (1954). *The theory of algorithms*. USSR Moscow: Leningrad.

Page, S. (2012). *Chapter 5 - Markov processes*. Retrieved from https://people.math.osu.edu/husen.1/teaching/571/markov_1.pdf

Taube, H. K. (2004). *Notes from the Metalevel: An introduction to computer composition (studies on new music research)*. New York, NY: Taylor & Francis. Retrieved from http://www.moz.ac.at/~sem/lehre/lib/bib/software/cm/Notes_from_the_Metalevel/index.html