The Automatic Transcription of Music to Determine its Chord Progression

Anton Gouws

School of Electrical and Electronic Engineering
University of Western Australia

November 2012

Abstract

The worked performed by the author relates to identifying the cause and effect of noise within musical audio. A large pool of data was recorded by the author, and then analysed using current known techniques of audio analysis. Using software applications such as MATLAB [1] allowed for the accurate analysis of the data, which in turn allowed for accurate conclusions drawn from the data trends.

Due to the careful designing of the experimental work which allowed for the systematic introduction of noise into musical audio samples, the dominant noise sources that affect automatic transcription were found. Several major noise sources were identified from human error, computational time implications, frequency content, and instrumentation differences. Correctly identifying and analysing these sources then allowed for recommendations to the current automatic transcription process to be given which should see an improvement in overall accuracy (based on the results found in the experimental work).

These results were found to be significant as there has not been such a detailed and specific research project performed (at least publically). This would allow future research into this subject to be more guided as how to improve the overall accuracy of the complete automatic transcription process.

ACKNOWLEDGEMENTS

I don't feel that anything I write here can truly show my thanks to the people who have helped me through the last few months. I try to make it a point to thank those who have helped me complete not only this dissertation, but also my degree, in person. But there are always those who have not do anything specific regarding my work here, but have helped more than they can imagine.

To my friends who has made my education into more than an educational experience. To my family who has never shown me anything by unconditional love and support. To those who have intellectually stimulated me, given me opportunity to argue endlessly over nothing, or simply listened while I explain my work even though we both know they will never understand it. Thanks.

Also a special thanks to Sarah for not only reading my whole dissertation, but actually giving me honest advice about it.

And lastly to the most important person I need to thank…

INDEX

LIST OF TABLES

LIST OF FIGURES

1 PROJECT SUMMARY

This project deals with the automatic processing of musical audio to determine its musical properties, with particular focus on musical chords.

Current methods for accurately extracting musical properties of a song require human experts to apply years of experience, and even then it is prone to errors (albeit minimal depending on the expert). With advances in technology allowing even the amateur artist to record professional sounding songs within their own home, the number of songs that has emerged into the market (also due to the ease at which new artists can enter the market through mediums such as Spotify[1] or Soundcloud[2]) has increased exponentially over the past few years.

The automatic transcription process is still far from being considered a perfect method, where noise sources in the data can often contribute significant errors to the final transcription, and therefore is vital to understand where these noise elements originate from. As there is not sufficient information on the sources of noise within a musical audio recording, the aim of this thesis is to investigate these and apply to the current method of automatic transcription to improve its accuracy. After conducting a review on the previous work on this topic to understand what methods are being used within the transcription process, an experiment was designed to identify the major noise components within a final recording.

Through the careful introduction of the suspected noise sources by using different instruments and recording techniques to obtain data sets of 48 chords, a detailed picture of what audio looks like in a data context was found. These were then analysed using a simple method of automatic transcription, the noise sources and their effect on the transcription accuracy were identified, and the source of these errors identified. To conclude the experimental work, a sample full song was recorded and analysed to observe the effects of the noise identified in the previous chapters where valid. These finding were then related to the current methods of automatic transcription, and suggestions put forth to improve the current automatic transcription process. This detailed study can be used by future individuals wishing to improve the accuracy of the automatic transcription of music, as sufficient data and analysis are present to apply to current methods of automatic music transcription.

[1] http://www.spotify.com/

[2] http://soundcloud.com

INTRODUCTION

1.1 PROBLEM DEFINITION

Current methods of automatic music transcription see an accuracy level of around 85% (achieved by Matthew Mauch [2]), and this is after many years of research into the subject. The complex and variability of the data to which the process is designed for causes many inaccuracies in the final transcription output. Finding the sources of these inaccuracies, namely what aspects of the data are defined as noise in the context of automatic transcription would help bring the accuracy of this process to a near perfect level. This means that a method of identifying the source and effect of all the relative noise sources would help guide future research into this topic into the right direction.

The study of different instruments and their effect of musical data will also be analysed, with particular focus on how the data differs between instruments and how this difference could interfere with the standard model of automatic transcription. This is an important aspect in the context of this subject, as there is a large mixture with no particular standard to what instrumentation is included in songs. This can cause a large amount of noise within the audio data which leads to incorrect transcription, but with no real indication as to why the error had occurred.

Due to the number of different aspects that can be studied in the improvement of this technology, it is important that an accurate method is found to collect, analyse, and interpret the data used within the experiments performed. If an accurate process can be developed in the future, many benefits will be seen across many fields of study. These includes benefits to social scientists who can identify why different populations listen to different music, other scientists who can more closely study the change in music over time, engineers who can modify this process to extract data from a large variable data set, and even the amateur artists who would gain access to accurate music transcription. There are many more, which is why the development of this topic is an important part of human development.

1.2 INTENDED AUDIENCE

The typical audience of this thesis would be individuals with at least an engineering background, but not necessarily a musical understanding. Although at least some knowledge is assumed on what constitutes as a 'song', which is the main data considered when applying the automatic transcription process. In-depth definition of the musical terms used throughout this thesis is outlined in the literature review, and enough information is given to understand the thesis as a whole without having to research musical terminology and theories separately. This is done such that the reader is not left in the dark, which is a possibility given the small likelihood that an in-depth knowledge of music, musical trends, and other aspects are known.

1.3 LAYOUT OF THESIS

Chapter 2 of this thesis outlines the previous work on the topic of automatic transcription, and identifies the required process which will allow for the experimental work to use a 'standard' method of automatic transcription. This chapter will be broken into several sections relating to the most important aspects of an automatic process.

Chapter 3 and 4 outline the concept and detailed design of the experiment. The main focus is on the methods of data collection, how the data samples will be varied, and the overall design of the experimentation process. Chapter 4 especially gives a really detailed look into what methods were used for recording the audio itself, such that any future reader of this project can understand the methods used and the assumptions behind them.

Chapter 5 shows the experimental work itself, giving samples of the collected data and a small analysis on the data trends observed. This is then analysed in detail in chapter 6 which is related to the findings and discussions. Here the different noise sources were identified and the reason for their existence in the data samples itself. This is then wrapped up in chapter 7 which gives recommendations to the current methods of automatic transcription. These recommendations are based on the data collected which would show an increase in transcription accuracy, which is the main focus of this thesis.

2 LITERATURE REVIEW

This literature review will be done in four distinct sections. The first outlines any relevant knowledge of musical theory and terminology required to comprehend the remainder of the thesis. The second part details how the data is represented throughout the process, which is an important concept to understand as the whole project centres around the analysis of this data. The third aspect is how the data is extracted from a musical audio file, with particular focus on the theoretical aspects as programs such as MATLAB can easily handle the practical implementation of this theory. The fourth aspect is how to manipulate the data extracted in such a way to obtain the outcome desired, namely the underlying chords in a song as they change over time.

2.1 MUSICAL THEORY BACKGROUND

To first understand the work done in the field of automatic chord transcription, it is important that the music terminology and theories used throughout the process is clearly defined. There are several concepts which overarch the entire process, and starts with what the goal of music transcription is: identifying a musical chord in time. But to understand this meaning, we must first define a chord. And to first define a chord, we must first define a note. And to define a note, we must first define pitch class and pitch height. So we start from the beginning.

Pitch is defined by Anssi Klapuri [3] as:

Pitch is a perceptual attribute which allows the ordering of sounds on a frequency-related scale extending from high to low

To mathematically define all relevant pitch classes (or notes), we first define a reference frequency which is usually ‘A’ above middle ‘C’. It should be noted that pitch class and note will be used interchangeably throughout this thesis, where later work refers to this simply as ‘note’. This reference frequency can range from 415 Hz to 445 Hz, but is usually 440 Hz when considering the standard pitch tuning in western music. An octave of a specific frequency can be simply described as the doubling of that frequency, where the range of frequencies that cover a single octave contains 12 distinctive ‘pitch classes’. To calculate the frequencies of different notes, we can use the following discrete series working from the reference frequency:

$$f_p = (f_{p-1}) \times 2^{1/12}$$

Where f_p is the frequency of the next pitch class (for example a reference of A=440 Hz, the next frequency will be A# or 'A sharp' of about 466 Hz).

Pitch height can be defined if we consider a single octave. An octave note is a note with the same pitch class, but at a frequency double the non-octave equivalent. In a musical context, we can define both frequencies of 440 Hz and 880 Hz as the note of 'A', but at different pitch heights. Each pitch class is defined by the English alphabet as a letter ranging from A to G, with some having the notation of a 'sharp' (denoted by '#') which in itself defines a new note. This gives each of the 12 pitch classes contained within an octave a common method of reference, and is adopted as the standard in popular western music. Since it is not intuitive to which letters also have a sharp notation, it is simply easier to list them in order starting with the class 'A', being:

[A, A#, B, C, C#, D, D#, E, F, F#, G, G#,]

A naming convention here denotes two subsequent notes as a 'semi tone' apart, meaning that an octave consists of 12 semi-tones. This can be represented by the following table which also shows the relevant frequencies of each note (with reference of A = 440Hz):

TABLE 1 - FREQUENCY LIST FOR EACH NOTE WITHIN AN OCTAVE

Note	A	A#	B	C	C#	D	D#	E	F	F#	G	G#	A
Frequency (Hz)	440	466	494	523	554	587	622	659	698	740	784	830	880

The convention of having an octave note denoted as the same letter makes sense in a musical context, as humans often interpret octave notes as equivalent [4], and therefore interpret chords regardless of pitch height. Next we must define the most important concept in this thesis, namely the chord.

The Virginia Tech Multimedia Music Dictionary defines a chord as follows [5]:

***Chord**: The sounding of two or more notes (usually at least three) simultaneously*

This definition will overlay the quest to define a chord from musical audio as a chord will only be considered as two or more notes sounded together. This in contrast to the human ability to 'perform successive interval abstraction' [6], which means that an arrangement of notes played separately, but within a musical interval (this depends on the timing signature of the music being played) can still be extrapolated as chords. This is similar to the human ability to fill in missing notes from a chord (due to what is known as harmonic memory) which means that the human interpretation of a chord could contradict the strict definition of a chord. These concepts will not be assessed in this thesis, and only chords as per the definition above will be considered (to aid in the possibility of constructing a real time analysis system).

Another important concept to define in the context of this thesis is the notation used to describe chords in a text format. There is great variety in how popular music is notated, especially after the boom of online music tablature sharing such as Ultimate-Guitar[3]. A review of different ways chord labelling is done was conducted by various authors [7], and shows what is common in almost all notations. Typically a chord is described by its root note (the lowest note in the set of notes making up the chord) and a chord quality. The root note is notated by the equivalent pitch class identifier (example A or B#) then followed by the chord quality text equivalent.

A chord quality is a method of describing how to construct a chord when a root note is given. A major chord (denoted by Maj) is constructed using the 'root' note (simply the lowest frequency of the three note chord), the 'third' (4 notes above the root note), and the 'perfect fifth' (7 notes above the root). The definition of a 'major third' is related to the traditional transcription of musical notes on a musical staff, and described the 'third' and 'perfect fifth' being three and five staff positions above the root note. Pitch class is the only important factor here, and pitch height does not dictate the chord. The following table shows a few examples of the types of chord classes used in music, where each type has a standard method of constructing the chord given the root note:

TABLE 2 -DIFFERENT CHORD TYPES AND TEXT EQUIVALENT

Quality name	Major	Minor	Diminished triad	Augmented triad	Seventh	Suspended 4th
Text equivalent	Maj	Min	Dim	Aug	7	sus4

This notation can be interpreted by humans and computers alike, and is therefore adopted by

[3] www.Ultimate-Guitar.com

the ISMIR - The International Society for Music Information Retrieval - as the standard notation. The concepts listed above are the building blocks for all following research and literature in the subject of automatic chord transcription. With this basic knowledge on music terminology and definitions, previous work in the topic of automatic transcription can be described.

2.2 LITERATURE OF PREVIOUS RESEARCH IN THIS SUBJECT

Due to recent popularity in this subject, and also improvements in hardware ability, there has been a lot of advancement made in the field of automatic chord transcription. The first extensive research into the subject can be traced back to Takuya Fujishima [8]. His research used a Fast Fourier Transform, a 12-vector chord mapping model as its lower level processing, a "chord change sensing" higher level model to further increase the usefulness of his method, and allowed the detection of up to 324 chords (considerably more than most other research in the area). This research was a substantial starting point for the rest that was to follow.

The process of automatic chord transcription can be broken into several independent processes and methods that need to be constructed (with some dependencies) which can then perform the task from raw audio to the final transcribed chords. Chapter 2.2.1 deals with how to model chords in such a way that a computing algorithm can compare the extracted data to. Chapter 2.2.2 deals with how to extract the audio data to a point where it can be evaluated, and chapter 2.2.4 deals with how to interpret this data to obtain chord-information.

2.2.1 CHORD DATA REPRESENTATION

There are two apparent ways of representing chords in a data format, each with their own advantages and disadvantages, but both doing essentially the same task. We will only consider the methods applicable (or apply the methods to) the western musical 12 note per octave scale.

The first method was presented by J. W. Ulrich [9] where he represented a chord by specifying the root bass note, and the subsequent notes as 'distances' from the root. An example would be the D-major chord which consists of notes [D, F#, A]. In this format it would be represented by the notation D (4, 7). Here F# is considered to be 4 semi-tones above D, and A is equivalent to 7 semitones above the root note D. This method allowed Ulrich to represent what is known as 'extended jazz' chords which contain notes more than 12 semitones apart. This method uses referenced data, where all relevant notes within a chord needs to be interpreted with respect to the provided root note and should be considered when notes within the chord are more than 12 notes apart (which does not strictly comply with our definition of chords). This is however not the case in academic approaches to the task of automatic chord transcription where ‘octave equivalence’ is assumed.

The second method was introduced by B. Pardo [10] where he represents a chord based solely on pitch classes (discarding pitch height). Here he defined the note C as the number ‘0’, and each subsequent semitone would be represented by a single integer higher number. This means we can represent the ‘D-Major’ chord simply as <2, 6, 9> (for [D, F#, A]). This method allows for a more standardised approach where extracted information from music can be more easily compared to a ‘chord library’, which is a database containing all the chords to be considered when trying to match the audio domain to the musical chord representation.

A more suitable method is representing a chord in the form of a 12-dimensional vector. Takuya Fujishima [8] method of chord abstraction left the data in a 12-dimensional vector with each dimension representing a semi-tone. This means each dimension represents a ‘note’ and can be described in either binary representation (1 or 0) [8] or a decimal [11] (which would represent the relative power of that frequency), but in a purely representational format, binary is used. The representation of the D-Major chord (consisting of [D, F#, A]) with the first dimension of the vector representing the semi-tone of C would be [0,0,1,0,0,0,1,0,0,1,0,0]. This method again assumes octave equivalence of notes, and has the advantage of simple comparison techniques to other chords through matrix multiplication.

2.2.2 Audio Data Extraction

The task of obtaining useable data from real world audio can be quiet complex (especially with additional implications brought upon by real world music audio files), and is dealt with by using common and well known techniques in signal processing.

The first method used by T. Fujishima [8] to obtain the featured extraction was through the discrete Fourier transform (DFT). Here the spectrum X_k is calculated using the following formula:

$$X_k = \sum_{n=0}^{N_F-1} x_n \omega_n e^{-\frac{2\pi i}{N_F}kn}, \quad k = 0,1,2,\dots,N_F - 1$$

Where N_F is the number of samples within one frame, ω_n is a window function that weights the samples to reduce unwanted effects such as spectral leakage (which is associated with finite observation intervals [12]).

The next stage was to assign each pitch class (from C to B) to the value $p = 0, \ldots, 11$ which is then used to calculate the PCP value. The PCP (which relates to the PCP Characterization Theorem) value is calculated through the sum of all power spectrum coefficients closest to an instance of that pitch class (this is how octave equivalence is implemented in the practical application of his method).

$$PCP_p = \sum_{M(m)=p} \left\|X_m\right\|^2$$

Where

$$M(m) = round(12 \log_2(\frac{f_s}{f_{ref}} \times \frac{m}{N_F})) \text{ mod } 12,$$

Where f_{ref} is the reference frequency of pitch class $p = 0$, and f_s is the sample frequency. Usually, the spectrum is calculated on overlapping frames over the duration of a piece of music in a process called *short-time Fourier transform* (STFT). This process leaves behind a matrix ($X_{k,m}$) in which the m^{th} frame occupies the m^{th} column and is called a spectrogram.

In the way a spectrogram describes the spectral content over a sample of time, the chromagram matrix ($PCP_{p,m}$) describes the 'chroma', or note content over a sample of time. Here the chroma vector of the m^{th} frame occupies the m^{th} column (If it a chromagram abstraction is performed over time). This 12-dimensional vector describes the power content of each relative frequency relating to each semi-tone class over time, and is the base data used for all subsequent calculations. Although different methods of abstraction can differ, the final product is always this 12-dimensional chromagram. To get an accurate representation of the current spectral content can be difficult for several reasons, the first being that a simple frequency spectrum does not translate into what is actually present, as an instrument will show a high power reading at the frequency being played, but also at related frequencies (known as upper partials). This is due to the mechanics of most instruments, especially stringed, wind, and bell classed instruments [13].

Other methods adopted by other authors to get around certain problem areas (such as the varying or unknown reference frequency to which all relating notes are tuned to in an instrument) was done by E. Gomez [14] where she detects spectral peaks first and maps only those to the chroma vector, according to their calculated frequency position.

An alternative strategy to obtaining the chromagram is to perform a log-frequency transform first and then assign the different pitch bins to the appropriate pitch class bin (there are 12 bins per octave in western music). This is because from the equation to calculate successive pitch classes, it is obvious that pitch is linear in a logarithmic scale. The most common method of this is to apply a constant Q transform [15]. Here a number,

$$Q = \frac{f}{\delta f}$$

Is the ratio of note frequency and note bandwidth (which defines the range of frequencies a note will be defined as, usually less than one Hertz). Now given the desired number of bins per octave (n_{bin}), we can calculate Q such that

$$Q = \frac{n_{bin}}{\ln 2}$$

The calculation of the constant-Q transform in the time domain involves separate windowing for every constant-Q bin, but equivalent windowing in the frequency domain can be performed simply (which requires a simple matrix multiplication of a kernel matrix with the discrete Fourier transform spectrum). Several authors have utilised the constant-Q transform in their method of chord extraction [16] [17] [18] [19]. The main downfall of this method is that the constant-Q transform requires a very long frame size for low notes, and short windows for higher frequencies can result in part of the signal not being considered in the extraction process unless the hop-size is set to a very low value.

For the convenience of the work in this thesis, a MATLAB toolbox will be used which performs the above methods leaving behind a 12-vector chromagram that can be used in the process. The toolbox was developed by several authors working at the department of music at the University of Jyvaskyla [20]. This toolbox contains original functions by the author relating the retrieval of information from music, and should prove extremely useful.

2.2.3 REAL WORLD SIGNAL PROCESSING IMPLICATIONS

Since the extraction of music is only a valid task if it is applied to real world signals, certain implications can be experienced when trying to build a working real world model.

The first consideration is speed of processing. This is overcome by the implementing the discrete Fourier transform using the fast Fourier transform (FFT) process [21], which drastically speeds up the calculation of the frequency transform. This method does impose some limitations, being the frame length N_F can only be a power of 2. Given a sample frequency f_s, the larger N_F that is used, the better the frequency resolution whilst implementing a smaller N_F would result in a better time resolution. This limitation needs to be considered when analysing the lower frequency spectrum of the signal, as it can become difficult to resolve simultaneous sinusoids at lower frequencies. Let's consider the following example:

$$f_s = 11025\,Hz, \qquad F_S = 2^{12} = 4096$$

Then the distance between frequency 'bins' or range of analogue frequencies which is considered to be equivalent in the digital domain would be

$$\frac{f_s}{F_S} = 2.69\,Hz$$

For the notes of E1 (41.2034 Hz) and F1 (43.6535 Hz) which have a difference of 2.45 Hz, this difference is less than the bin size. This could cause a problem when considering methods of automatic chord transcription suggested by [2] where he considers the bass and treble spectrum separately to increase the accuracy of transcription, and realising the importance bass notes play in the classification of chords. This in a musical sense however does not cause too much of a problem, since such closely spaced bass notes do not occur simultaneously. But it should be noted that the ability to resolve sinusoids depends largely on the window function used.

Window functions are extremely vital to the Fourier analysis of finite signals, as they determine both the sinusoidal resolution and noise robustness. Author F. J. Harris [12] compares 23 different window functions in terms of their theoretical properties. He noted that there is always a trade-off between main lobe width (which should be narrow for higher sinusoidal resolution) and side lobe height (which should be low for better noise robustness). This trade-off means that no single window function can be claimed as the best choice without considering the data it is to be used with first. This in parallel with the fact that music is not monotone in its data characteristics, no single window function can be claimed as the best choice for all music. Different authors seem to use different window function without giving much reasoning, whilst others omit that data completely. The most notable authors who compared the functionality with different window function were [22] who compared 3 different window functions, and although the difference between them were minimal, found that the Blackman window worked best. Other authors such as [23] used the Blackman-Harris window, [24] used the Hamming window. Most noted authors use the Hamming window, but this could be due to it being the default window in the MATLAB spectrogram function.

Other real world considerations are noise (where noise here relates to any data included that does not relate to the underlying chord detection). There is no detailed study as to what noise is contained within a piece of music, and their effect on automatic transcription. Some authors however do note techniques used to reduce certain types of noise such as percussion. One such method used to reduce percussion interference is to 'smooth' the chromagram in the time direction using a finite impulse response low pass filter [17]. Another method involves subtracting the background spectrum (a smooth noise envelope which is calculated by 'median smoothing' of the spectrum) from the spectral representation [25]. But not much detail is provided in most literature which details the noise of an audio file relating to transcription by an automated process.

A different approach to obtaining the chromagram is through 'beat tracking' which notes that music always plays in 'beats' or repeating time periods where chords seem to change in time with these [26]. What this method entails is to average out the spectrum over a single music beat, and use the subsequent information to calculate the chords. This has a welcome side effect of noise reduction as percussion usually has a low spectrum content over time, but high power readings at periodic instances (similar to an impulse function). Using this method can help diminish the effects of percussion, but has several limitations including the assumption that chords do not change over a single music beat (often not an accurate assumption), and has the added task of automatic beat recognition which has been proven to be a problem (as noted by several authors including [27], and [28]).

One of the biggest problems in trying to use the data acquired to try and match it to chords from theory is that musical instruments often introduce what is known as 'overtones', which are harmonic frequencies which are integer multiplications of the fundamental frequency [3].

Methods used to reduce harmonic overtones include a method developed by [14]. Here a weighting was given to each i^{th} harmonic to emphasise the fundamental harmonic (which all subsequent harmonics are integer multiplications). Here the weighting of the harmonics decreased exponentially such that each subsequent i^{th} harmonic contributes s^{i-1} of its energy, where Gomez chose $s = 0.6$ (s is a manually tunes number)

The last issue to deal with in real-world application is tuning. Since the tuning of an instrument can range depending on the frequency chosen for its reference note (usually A above middle C which is tuned to 440 Hz, but can range from 415 Hz to 445 Hz), an automated program can assume a specific tuning, pick off the harmonic data based on that even though the actual tuning is different resulting in an error over the whole spectrum data. For a reliable chord transcription process, this tuning has to be determined by the program itself to account for the possible range amongst songs, as a standalone program would not be practical to manually enter this information. Methods implemented include using 36 chroma bins, 3 per pitch class. The program would then determine which chroma (out of the three) has the highest power reading, and estimate this as the tuning of the song [17]. It is fairly practical to assume a tuning reference of 440Hz however, as this is used in almost all popular songs (and is the standard setting for electronic tuners).

2.2.4 DATA PROCESSING

Once the 12-vector chromagram is obtained, how the data is processed is the next important step in the automatic transcription of chords. The best aspect of using the 12-vector chromagram approach is that simple matrix multiplication can be used, which is easily implemented with programs such as MATLAB.

Since it is possible to build a chord profile manually using the 12-dimensional vector with each vector representing a pitch class, the first step is to build a large enough database of pre-defined chords in this format which all data will be compared to. It would be intuitive to build the largest possible pre-defined chord data set, such as T. Fujishima [8] who had 324 chords, but this could actually lead to more inaccuracies. Since previously it was mentioned the amount of 'noise' sources present in music (especially live performances which doesn't have the benefit of noise removal that professional studios have), the larger sample set you have, smaller difference between two separate chords. As a single note can alter the meaning of a chord in the more complex chord identifiers such as Suspended-4th chords, high enough noise can push the interpretation away from the true chord.

Most methods utilize anything from 24 chords [29] to 204 chords [30], with the average being 48 chords [31] which seems to cover the largest range (Major, Minor, Augmented triad, and Diminished triad chords) without introducing unnecessary errors (as most music compositions do not stray very far from the Major and Minor chord structures – with the exception of jazz or other instrumental based genres). This data base is then used to compare the real time data in various forms.

The simplest method was used by T. Fujishima where he simply takes the inner product of the instantaneous chromagram with the pre-defined chord database. Simple matrix multiplication means the evaluation is quick, and gives the final data in simple to interpret form. To show this method of transcription, the chord G-major chord is transcribed below. The process would be along the lines of transposing the data-based chord matrix, multiplying the two matrices together, evaluating the resulting single-entry matrix (the higher the number, the better match to the chord it is), then proceed to do so for each of the pre-defined chords, picking whichever results in the highest match. From this it is clear that the more pre-defined chords that are given; the longer the process will take.

Comparing G-Major with several similar chords shows the following results (using theoretical binary values):

TABLE 3 - DIFFERENT CORRELATIONS WITH G MAJOR

Chord	Pitch Classes	Inner Product
G major	B, D, G	3
C major	C, E, G	1
G minor	A#, D, G	2
G Diminished triad	A#, C#, G	1
G Augmented triad	B, D#, G	2

This shows that even though the G-Major chord shows similarities with other chord classes, the highest inner product is with that of the G-Major of 3. The advantage of this method is that power readings can be substituted for binary values. This means values between 0 and 1 can be used in the vectors to improve the accuracy. It can be shown (due to overtones and other reasons) that the power readings for a chord does not always give the same power reading for two pitch classes even though they are played simultaneously. This gives the option to 'learn' what chords look like [32] from a structured learning exercise, or use the theoretical binary values as Fujishima did.

It has been shown how a single Gaussian can be used to estimate not only chords but the key of the song as well, it has not been the best method utilised with a lot of research currently going into finding the best method for 'profile matching' (matching up the data with the theoretical meanings). Other methods besides Gaussian include using Dirichlet distributions [33], Neural networking [34] [11], and learned feature models which were automatically taught [35]. It should be noted that most methods learn relatively few chord data profiles (usually 24) as it has been noted, especially due to the surprisingly low performance of a model which uses many chord types [32] suggest that there is an optimum amount of chords to which the data is compared to which allows for the best overall transcription accuracy in a global sense.

2.3 CONCLUSION

It is obvious that one thing omitted from most literature studies is a detailed analysis on the noise content that could be contained in a musical audio sample. Since the overarching goal of this thesis is to try and improve the current process in some aspect, it is counterintuitive to try and do this without a good knowledge on the data itself.

Methods for collecting the data, manipulating the data, and finally achieving the goal of automatic transcription are possible, but its accuracy is nowhere near a perfect system yet. Previous work has been focussed on achieving the actual final result in any form, this has been achieved over the past few years. Future focus into the area should be at improving the accuracy of transcription beyond the 85% mark, and this can only be done by understanding the noise sources and their effect within musical audio.

This will be the basis of all research in this report, where an experiment will be designed to analyse various sources of musical audio, and identifying the sources of noise relating to automatic transcription. Having an accurate knowledge on what factors in a song actually contribute to less than perfect transcription rates is ideal to give realisable suggestions to the current process of automatic music transcription. With this in mind, an experiment can be designed to identify the cause and effect of noise in the context of automatic musical transcription.

3 CONCEPT DESIGN

The aim of the experimental work within this thesis is to identify the different sources of noise within a piece of musical audio, and also identify their relative effect to the final transcription of music. The experiment will be designed such to systematically introduce the possible noise sources into a data set. This is then analysed and compared to demonstrate the effects. Several different audio sources will be used to generate the same data set which will allow for the precise control of audio spectral content. This concept of recording and analysing the different audio data sets can be broken down into the following steps:

FIGURE 1 - CONCEPTUAL METHOD FOR DATA EXTRACTION

The first block relates to the individual instrument, which will be recorded in isolation. Each individual chord (48 different ones covering the Major, Minor, Augmented triad, and Diminished triad voicing's) will be recorded and put through the chromagram algorithm which identifies the spectral content in musical notation (depicting notes instead of frequency). This will then be put into a database, where each different data set will be compared and analysed. This process can be further broken down into their respective details, which results in the more complex block diagram:

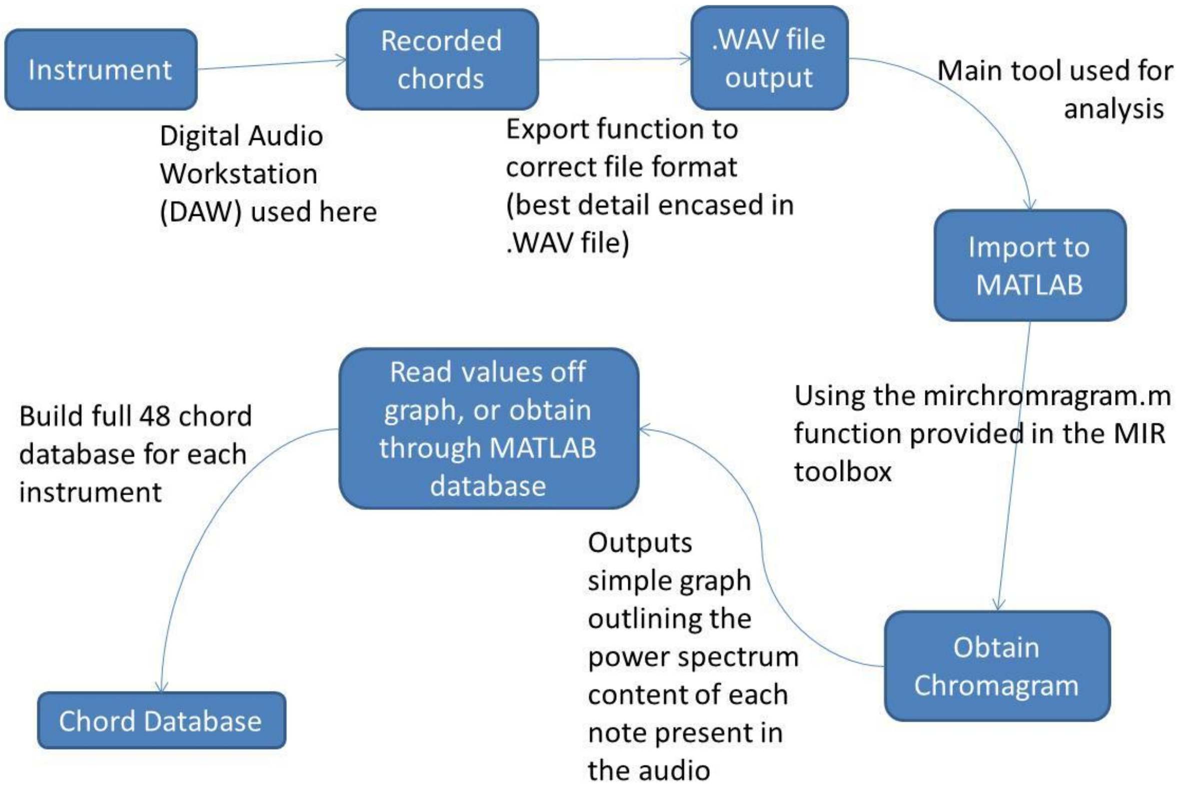

FIGURE 2 –DETAILED DESIGN OF ACQUIRING THE DATA

For the recording of the audio, there are several different options that can be explored. The first is to obtain different sections of pre-existing songs where it has been transcribed by a professional, and segment the chords from that. This method does utilise real data which the process will be applied to, but if multiple sources of noise is present within the sample, they cannot be separated and identified correctly. The best solution here is to record the data specifically for this thesis, which will allow for the systematic addition of noise. By this concept, it should be possible to introduce noise in such a way that when compared to the previous data set (which will contain all the noise content found to that point), the effect of the noise being currently analysed can be easily deduced.

The chords covered in this thesis will be the Major, Minor, Augmented triad, and diminished triads as these will cover the all main chord types used within music. More complicated chords are not included, as doing so would not bring forth any more clarity to how the noise can affect the audio samples. In terms of instruments used, this will start with computer generated sound to be used as a control. It will then move into human played instruments, where different instruments and recording techniques will be used to help identify the noise content.

In conclusion to the data to be analysed, a small sample (including mainly major and minor chords) of chords will be analysed from musical audio containing other musical elements such as percussion, bass, vocals, and effects. This data set will emulate real world data to which the automatic chord transcription process is designed to analyse. It is therefore important to this whole thesis that the behaviour of these results can be related to its individual components, each of which can be isolated from the progressive introduction of all error elements commonly found in the final data set.

It should be noted that the method of determining if a noise source affects the final transcription is where a note is correctly or incorrectly transcribed. Therefore a simple maximum likelihood algorithm will be written to actually transcribe the chords recorded. This would not be the most efficient method of transcribing the chords (and also providing additional information needed for comparison between data sets), but it will represent the most common and easiest method of transcription.

4 DETAIL DESIGN

The detailed design of the experiment will outline the more specific aspects of how the information is obtained and handled. Working on the conceptual designed depicted in figure 2, there are several stages in the process, with additional and specific information added to each. They are broken down into the subheadings describing each function as dictated in the concept design block diagram.

4.1 INSTRUMENTS

A final musical recording can contain many different instruments recorded in many different fashions. To correctly identify the source and effect of a single noise source, it is important to use instruments that not only represent a realistic example of what can be used in a song, but also ones which allow the introduction of noise systemically. For this reason several different instruments were chosen to record the 48-chord data set, where each one adds a single source of noise (such that as you progress through the experiments, the current audio samples contain the same errors as previous, but also the new one being identified). The first instruments, which will be the control, are computer generated sine waves. This allows for no variation in note volume, timing, and instrumentation noise. This will be generated inside a digital audio workstation (DAW) which is the method that would be used in actual professional recordings. This will identify any errors in the computation techniques as they should theoretically represent the zero-noise chords.

The next instrument to be used is a piano synthesizer. The reason for using this is to introduce the effect of a real world instrument without the introduction of human error. Since the piano sound is constructed using real sample data recorded from a grand piano, the errors introduced by the instrument itself will appear superimposed to the previous errors found from the sine waves. Factors such as upper partial harmonics (frequencies present in a vibrating note that is not the fundamental frequency of the vibration occurring in the string) can introduce errors which would carry over into other stringed instruments such as guitars.

The next instruments to be used will be an electric and acoustic guitar. The electric guitar will be recorded and analysed first for a few important reasons. The recording technique itself is the main reason, as an electric guitar can be recorded via direct input. This uses the output of the magnetic pickups of the guitar, and feeds this directly into the analogue to digital (A2D) converter (which in this case is the sound card on the computing device used to capture the recorded sound data). This reduces other sources of error that can be introduced by the acoustic guitars recoding technique of a studio condenser microphone. This method also requires no effects commonly used in electric guitars be used to show the effect of a human player producing the chords.

Next will be an acoustic guitar, which brings in additional noise sources such as microphone recording. As this is the standard method of musically recording an acoustic guitar, it is important that the same standard method is used in recording of the samples. Orientation can affect the spectral content of an acoustic guitar (relative to the microphone), so the standard method of placing the microphone close to the acoustic guitar's sound hole will be used to represent a typical setup used in a studio.

A small side experiment will be performed on the electric guitar, where heavy distortion is added. This is a common and popular technique in such genres as heavy metal, punk rock, and many other genres. This experiment is primarily performed to prove a hypothesis on the effect of heavy distortion to the automatic transcription process. The hypothesis relies on two important factors brought upon by heavily distorting an electric guitar. The first is that it will bring the power content of each frequency being played by the instrument to the same level which would allow for simple binary comparison to the pre-determined chord data base. And the second is the limitation heavy distortion implies to the range of chords being played, where only 'power chords' are predominantly played (which consist of 3 notes, but the third note is an octave of the root note, so essentially only 2 different notes are contained within the chord) as Major/Minor chords either sound unclear when distorted, or they simply do not fit the musical styles of the genre. Both these factors would allow easier transcription of the final musical file

The different instruments used will be as follows, with extra detail added to the table for clarity:

TABLE 4 - DIFFERENT INSTRUMENTS USED WITHIN THE EXPERIMENTAL SETUP

Instrument	Recording source	Effects	File Output	Chords	Analyser
Sine Wave	Synthesizer	None	.WAV	48	Chromagram
Grand Piano	Synthesizer	None	.WAV	48	Chromagram
Electric guitar	Guitar Amplifier	None	.WAV	48	Chromagram
Electric guitar	Guitar Amplifier	Distortion	.WAV	6	Chromagram
Acoustic Guitar	Condenser microphone	None	.WAV	48	Chromagram

The full production song data will be recorded using an electric guitar, synthetic percussion (again a common method), an electric bass guitar, and a vocalist. Additional effects, such as compression and reverberation, will be added and represent another data set to more accurately represent a professionally produced song.

4.2 Recording Techniques

As previously mentioned, the method of recording the audio data can impart many errors in itself, and it is therefore important that not only an accurate method of recording is used for each instrument, but that they reflect the standard for the specified instrument (although given the wide range of methods that can be used, the method that will produce consistent data will be the preferred method). This section gives a more in-depth methodology for recording the sound across each instrument, including all the intermediate processes.

4.2.1 Sine wave generator

The particular generator used in this experiment is the ‘Nexus’ plugin produced by ‘reFX’ [36]. This is a popular tool used by professionals and produces high quality audio samples. The sine wave output gives a 44,000 KHz 24 bit generated waveform with no additional effects added for data integrity. The sound is generated inside the digital audio workstation (Cubase 5.1 [37]) and then exported into a .WAV file. There are no intermediate steps between the generated sound and the file output, giving the best representation of a chord constructed with 3 sinusoidal frequencies. Each note within the synthesizer will be set to the same volume and be played at exactly the same time (to avoid additional power being added to the spectrum from having a note being present for a longer period of time within the sample). This data set will be used as the control to which other results are compared to.

4.2.2 Synthetic piano

This method is fairly similar to the sine wave generator with one distinct difference, which relates to how the sound is generated. As mentioned previously, the synthetic grand piano sound is achieved by sampling individual grand piano notes and then reproducing them when specified by the user. Factors that can be controlled at this stage are the velocity at which the piano strings are struck (increasing the amplitude). This is achieved by not only adjusting the volume at which the sound is generated by the synthesiser, but by also recording each note at different volumes at the source. This is done to reproduce the different qualities introduced by striking the strings within the piano at different velocities. This method of producing a realistic piano sound that can be used in professional audio recordings allows for the best representation of a stringed instruments sound without adding the human source of error into it. Again, each note will be set to the same volume and played at the same time.

4.2.3 Electric guitar

The method for recording the electric guitar contains several additional intermediate stages when compared to the synthetic sound generators used in the previous section of the experiments. Here the most common method of recording an electric guitar is used, and the details provided here include every stage that is required for recording an electric guitar.

When the string of an electric guitar is struck, it vibrates the magnetic field (due to the ferrous conducting properties of the steel strings) around a coil of wire with a large number of loops. This creates an alternating electric signal in direct correlation to the frequency at which the string is vibrating [38]. This signal is then passed through the guitars electronics which consists of a potentiometer that controls the volume and a potentiometer and capacitor circuit that control the ‘tone’. This is simply a low pass filter, where the cut-off frequency is controlled by altering the resistor value in the RC circuit.

For the experiments, the volume control is set to full and the tone control is set to allow the maximum range of frequencies through the circuitry as to not deteriorate the sound quality. This signal is then passed through a mixer, which is simply an amplifying circuit that amplifies the low amplitude signal into a useable audio signal. Additional volume and tone controls are available on the mixer, where for the experiment the tone settings are ‘neutral’ which neither adds nor subtracts additional tonal qualities to the signal, and the volume is set such that no clipping is experienced in the next stage. This signal is then fed into a sound card which consists of an analogue to digital converter.

The sound card used here is an ‘M-audio Audiophile 2496 [39]’ which converts the analogue signal to a 24bit digital signal at a sampling rate of 48 KHz (studio quality recording). Again, no effects or tonal modifications are performed on the signal, and this is then exported into a ‘.WAV’ file. The guitar used for the experiment is a Gibson Les Paul guitar, which is of a high quality and is used extensively across many musical genres.

4.2.4 DISTORTED ELECTRIC GUITAR

The method for recording this will be done similarly to the electric guitar, except an intermediate gain stage will be added before the mixer stage. This will be done with an electric guitar amplifier, namely the ‘Line 6 Spider II’ model, where the high gain setting is applied. This will result in an effect similar to those found in genres such as heavy metal, rock, punk etc. No additional effects will be added.

4.2.5 ACOUSTIC GUITAR

The only difference in this method of recording in comparison to the method used for the electric guitar is how the audio signal from the instrument is captured. Here a studio condenser microphone is used (the standard method for acoustic instruments) to capture the sound from the guitar. The orientation of the guitar during recording is such that the ‘sound hole’, or more simply the opening of the body of the guitar where the sound resonates at its highest volume, is directly in front of the microphone at close range (within 10cm).

This allows for the capturing of all frequencies produced by the guitar, which in this case is a desirable feature in a musical context (often described as richness of the sound). The guitar used in this experiment is an Ibanez Art-wood acoustic guitar, again a fairly high quality instrument to reduce any noise associated with a lower-grade instruments (which would not be used in a professional production song).

4.2.6 FULL SONG RECORDING

For the final aspect of the experiment, a series of chords will be extracted from an audio sample which can be best described as a full production song. Here many aspects of a completed piece of music will be added to the audio. These aspects include synthetic percussion provided by a plugin called ‘Groove-agent ONE [40]’, which is provided in the ‘Cubase’ DAW package. The same theory as the synthetic piano is used to produce the sound, where samples of a real percussion kit are used to give an accurate representation of a studio percussion kit.

Another element added is a bass guitar, which will be recorded by the same technique as the electric guitar. Also included are the electric guitar and a vocal track. The vocal track will be recorded using the condenser microphone with no additional tonal quality added outside the recording software. Here two sound samples will be produced for every chord within the song. The first sample will be done with no additional effects added to the individual elements of the song. This is done as a control to see if adding effects to the elements will introduce notable differences in the data. The effects that will be used are as follows:

Compressor: This effect essentially amplifies lower volume portions of the audio, and compresses higher volume elements. This is done to bring the whole sample, to which the compressor is applied to, to a more consistent volume level. Here the effect is used in moderation such that it does not amplify the background noise of the recording to an audible level.

Reverberation: This effect adds a delayed sound onto the original audio track giving it a sound resembling a large hall. If you consider the delayed sound at the same volume of the original signal to be 100% effect added, here only 8% is used to add a desirable musical quality to the vocal and guitar track.

Equalization: Tonal modification will be used on the vocal and the guitar track. Here the modifications will be performed in such a way to separate the two sound files in terms of frequencies. A slight high pass effect will be used on the vocal track, and a slight low pass will be used on the guitar track. This is a common technique used which allows for the audible separation of different elements within a song, such that all the different aspects of a full production song do not get ‘tonally mixed’. This would result in the listener not being able to separate the different sounds within the final mix.

4.3 FILE OUTPUT

The file output used is the .WAV file, as this does not use audio compression which decreases the size of a file dramatically, but also decreases the audio quality. This is achieved by sampling the audio content at 16 bit rather than 24 bit (amplitude definition levels) and lowering the bit rate to (usually) 320 Kbit/sec. It is also required for analysis through MATLAB. The .WAV format is also utilised here given the wide level of adoption by other programs. For example, a default function in the powerful software package MATLAB (the main software tool used for analysis) called ‘wavread.m’ can be used which easily transcribes the audio data into matrix form. This data is then used by other functions such as the fast Fourier transform (fft.m) which allows for the representation of the data in the frequency and power domain. These functions are the basis to which an automatic transcription process is built on, as the data obtained through them is what is essentially manipulated into the final output.

4.4 Obtaining the Chromagram

As stated previously, the MATLAB program 'mirchromagram.m' [20] will be extensively used to obtain useable and readable data from the audio samples for several reasons. MIR stands for Music Information Retrieval, and is a very reliable source in the field of automatic chord transcription. The authors are regular attendees and contributors to the ISMIR (International Society of Music Information Retrieval) conferences where a majority of sources were obtained for the literature review, and a majority of the people who have significantly contributed to the field of automatic music transcription attend regularly to represent the current advancements in this topic.

After recording a single chord using a sine wave generator inside a digital audio workstation, the single chord was exported as a .WAV file and then analysed using the 'mirchromagram.m' function. The following figure was the output:

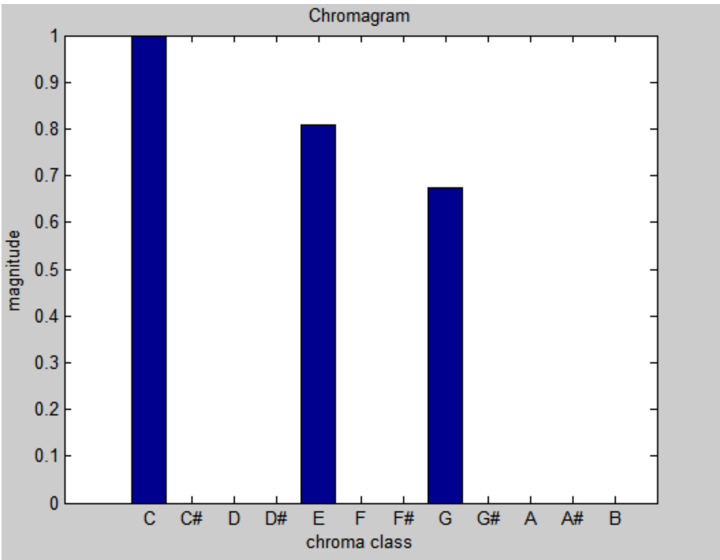

FIGURE 3 - C MAJOR CHORD CHROMAGRAM

It can be observed that 3 notes were present in the chord which matches with the theory (this particular sound file was generated using the Sine wave generator as stated previously). Here it is seen how the values are normalized such that the highest power value is shown to have a magnitude of 1 and the rest of the notes are scaled according to this note (C in this case, the root note). Using the 'data' cursor function within the figure display itself (within MATLAB), values can be read off and a data table can be constructed. This is the method that will be used to more closely analyse the data.

Since it is never a good methodology to use a program or function without independently confirming its output as the correct output, the functionality was verified via several manual steps and calculations. Using the knowledge that the frequencies played in the chord were generated using an external program used solely for music production (which suggests that the qualities present in the audio sample are those desired in a musical context), the generated waves should show frequency peaks at the known frequency values played in the frequency domain. It should be noted that the software that generated the sine waves was also independently verified by using an external hardware tuner, where a single note was generated then put through this musical tuner. The exact same note was displayed as the one generated and showed a degree of accuracy to 0.5% (as stated on the tuner's manufacturer's specifications) at least. This step merely confirms that the generated notes are as stated by the software used to generate them.

Manually working out the frequencies that should be contained within the signal is as follows:

TABLE 5 - C MAJOR FREQUENCY LISTING

Note	Frequency (Hz)
C	216.625
E	329.627
G	391.995

Writing a short MATLAB program (see appendix section 10.8) which computes the power spectrum of the sine C-major gives the following graph:

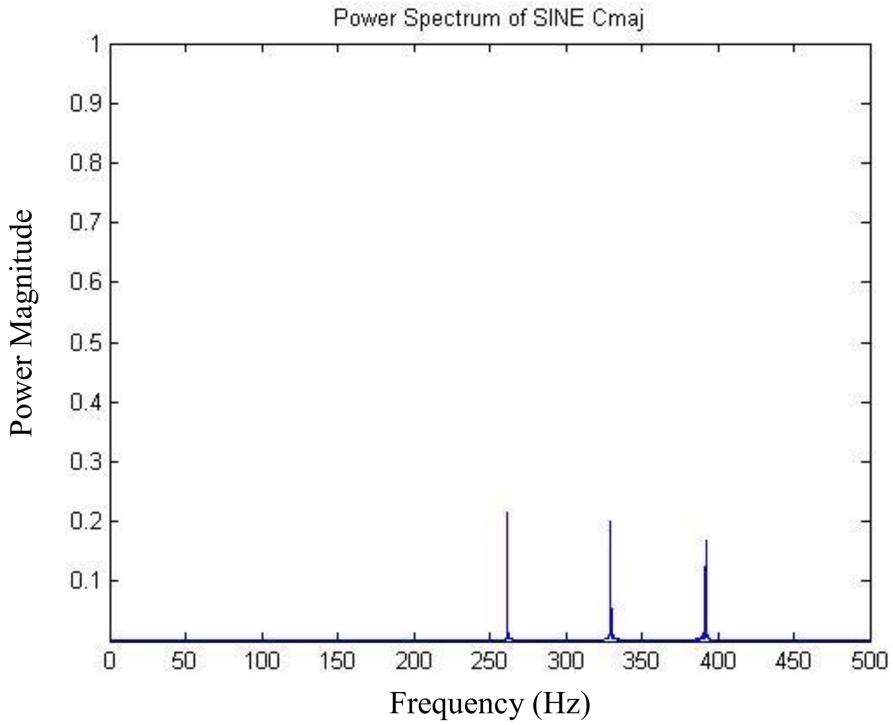

FIGURE 4 - POWER SPECTRUM OF SINE C-MAJOR CHORD

Which when zoomed in on the appropriate area gives the following graph:

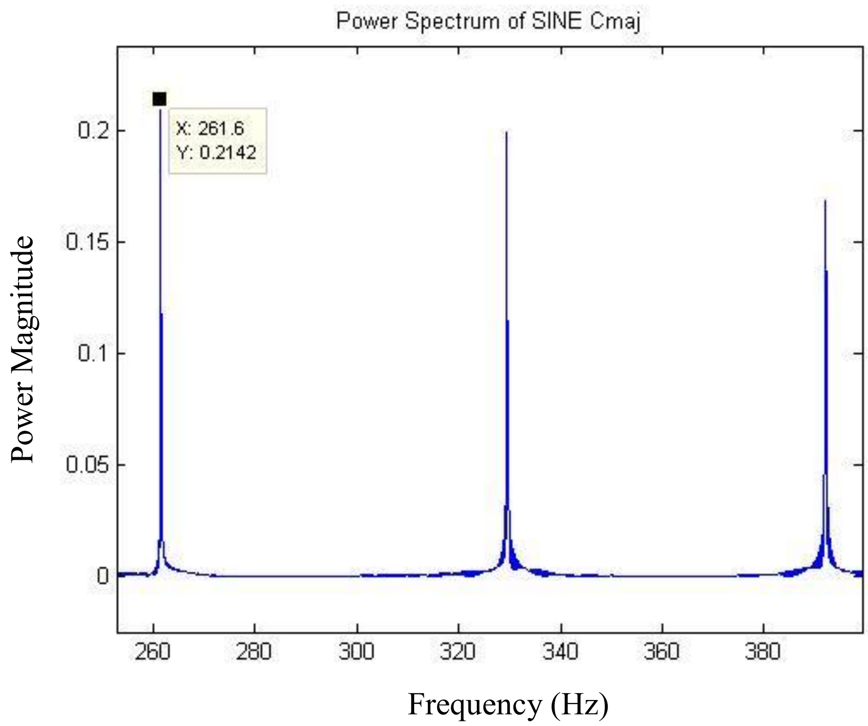

FIGURE 5 - POWER SPECTRUM OF SINE C-MAJOR CHORD

Which correspond to the following data:

TABLE 6 - EXTRACTED POWER VALUE OF C-MAJOR FREQUENCIES

Peak	Frequency (Hz)	Power
1	261.6	0.2142
2	329.6	0.1996
3	392	0.1684

This shows the correct correlation from frequencies to note value in the 'mirchromagram.m' function. It can also be observed from the spectrum that the power of each peak decreases as the frequency increases which also corresponds to the data trend shown in the chromagram. The short function written to obtain this graph simply used a fast Fourier transform, where as many data points was used that were allowed by MATLAB to ensure its accuracy. This gives enough confidence to use the 'mirchromagram.m' function in future analysis of all the audio signals.

4.5 Chord Database

The chord database representation will be shown in a simple table format as follows:

TABLE 7 - SAMPLE CHORD DATABASE FORMAT

MAJOR	C	C#	D	D#	E	F	F#	G	G#	A	A#	B
C	1				1			1				
C#		1				1			1			
D			1				1			1		
D#				1				1			1	

As evident above, the database has two axes. The left axis shows the chords being represented (with 4 major chords, from C to D# being shown here) and each chord type (major, minor, augmented, and diminished) set being made up of 12 chords. The top axis represents individual notes, again all 12 within the western musical scale.

The way in which a chord is represented is fairly simple. For each chord located on the left side of the table, each contributing note that is used to construct the chord is then represented as present (denoted by a 1 in this case) or not present (denoted by a blank, or a zero when transcribed to a vector which can be used by a function). So using an example from above, the C major chord is constructed using the notes of C, E, and G.

By using this format for constructing the chord database for each instrument, additional data such as each note's contributing power level to a chord can also be tabulated and used. In cases such as that, a binary 1 or 0 will not be used for a chord being present or not, but rather a decimal between 0 and 1 which relates to its relative power level.

Each chord relates specifically to what combination of notes are contained within it, where the major and minor scales see all 12 chords being unique based on the note content. Special consideration however needs to be taken for the augmented chords as technically there are only 4 different combinations for all 12 chords based on note content alone. For example, the D augmented contains the notes in chronological order of [D, F#,A#]. The F# augmented chord contains the notes [F#,A#,D] which match the previous chord. In this context, what separates these two chords will be the root note, which is the note of lowest frequency. All chords used in the experiments will be triads, which means they are made up of only 3 notes with the root note being the lowest frequency and the other notes played the least distance (in terms of frequency separation) required in relation to the root note.

4.6 Transcription Method

Once all the data has been tabulated with the above methodology, a transcription method is needed to observe how well the data in each experiment can be transcribed into useable chord information, and to compare accuracy rates between sample sets. The method that will be used in these experiments is the Maximum Likelihood algorithm. This method takes advantage of the matrix format all the data will be recorded in. A description of the method follows below.

A 12 element vector is formed containing the chromagram data of the audio file. The first entry corresponds to the note 'C', which is the standard starting point of all chord data banks. This is then pre-multiplied with the transverse of the data matrix containing the pre-defined chords. This returns a 12x1 matrix with correlation values. Given there are 3 notes in a single chord (assumed for this task), if a sampled chromagram contains these three notes at a power level of one, the correlation returned for this chord will be equal to 3. This means that the correlation values will range from 0-3, with the highest element giving the most likely estimated chord. A worked example for clarification is provided below:

MAJOR	C	C#	D	D#	E	F	F#	G	G#	A	A#	B
F#		0.67					1.00				0.77	

X

MAJOR	C	C#	D	D#	E	F	F#	G	G#	A	A#	B
C	1				1			1				
C#		1				1			1			
D			1				1			1		
D#				1				1			1	
E					1				1			1
F	1					1				1		
F#		1					1				1	
G			1					1				1
G#	1			1					1			
A		1			1					1		
A#			1			1					1	
B				1			1					1

T

FIGURE 6 - SAMPLE IMPLIMENTATION OF THE MLA

Where the F# Major chord sampled data from the Sine wave data set was used. When this matrix multiplication is performed, the following result was obtained:

MAJOR	C	C#	D	D#	E	F	F#	G	G#	A	A#	B
ESTIMATE	0.00	0.67	1.00	0.77	0.00	0.00	2.44	0.00	0.00	0.67	0.77	1.00

FIGURE 7 - MLA OUTPUT

The estimate in this case was the F# Major chord with a correlation of 2.44. This process is repeated for all four chord types where the highest value is returned. When the highest element is found to be equal for two different chord types (But same root note value, e.g. comparing A major to A minor chord), the Major chord is preferred, followed by Minor, Augmented, then Diminished in that order (due to the probability of use in music being that order in general [41]). The properties to be obtained from this analysis relating to the accuracy of transcription and the parameters of the data transcribed is the percentage of correct transcriptions, and also the average correlation values for correct transcriptions (where the highest correlation is preferred as it shows a higher probability of being a correct transcription, rather than a calculated guess).

5 EXPERIMENTAL WORK

In this section, a summary of the results obtained when applying the ‘mirchromagram.m’ function to each of the isolated chord audio files across each of the instruments can be found. A more in-depth analysis of the trends and accuracies observed from the data is found in chapter 6. The full set of data for each instrument (showing all the tabulated data obtained from the experimental work) can be found in the appendix section 10.2 to section 10.7. First the outline of the exact methodology used to obtain the data throughout this experimental work is detailed below.

5.1 METHODOLOGY

To correctly obtain and analyse the data, the following methodology was followed. Note this was developed through practical experimentation by the author of this thesis and then summarised in the steps below.

5.1.1 INSTRUMENTAL CHORD DATA COLLECTION

1. Record the 48 different chords using the recording techniques outlined in the detailed design for a specific instrument using the Cubase DAW.
2. Export the chords as an individual .WAV file spanning from 2.5 – 3 seconds (consistent for each instrument type).
3. Import the files into the MATLAB directory
4. Apply the ‘mirchromagram.m’ function to the audio sample.
5. Manually tabulate the power values from the provided graph.
6. Complete a data table for each chord type (4 in total) for specific instrument.
7. Repeat steps 1 to 6 for each additional instrument.

5.1.2 FULL PRODUCTION SONG DATA COLLECTION

1. Record a simple song with the chord progression as detailed previously.
2. Manually separate and export each individual chord as per BAR separation (one bar contains one chord, so each bar is exported separately)
3. Import the files into the MATLAB directory
4. Apply the ‘mirchromagram.m’ function to the audio sample.
5. Manually tabulate the power values from the provided graph.
6. Complete the table as shown in the appendix chapters 10.2-10.7.

5.1.3 TRANSCRIPTION

1. Use the function written by the author which implements the Maximum Likelihood algorithm.
2. For the 48 chord data set, values will be automatically returned as provided in the following sections (percentage correct transcription and average correlation).
3. For the full production song, compare the transcribed chords with the correct ones and manually tabulate percentage correct transcriptions.

So using this method of obtaining and analysing the data, the following experimental results were obtained for all the recorded data in this project.

5.2 SINE WAVE CHORDS

The audio samples relating the sinusoidal generated chords were analysed, where a typical chromagram returned a graph such as the one below:

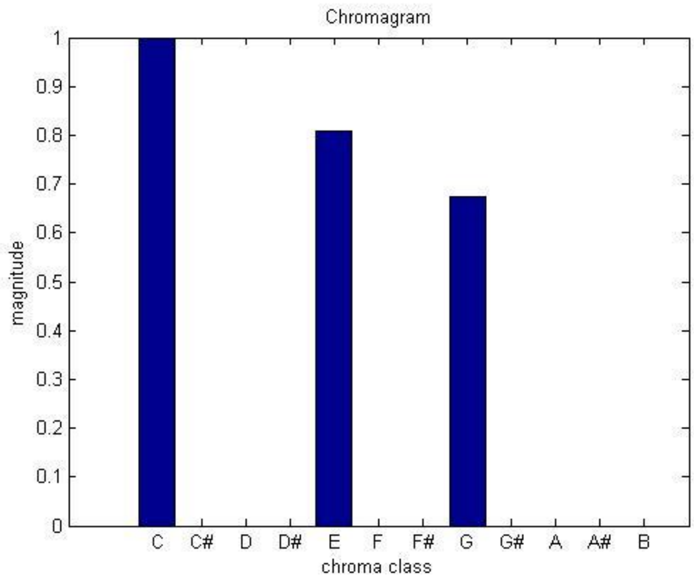

FIGURE 8 - SINE CHROMAGRAM FOR C-MAJOR

Not only is the trend of exponentially decreasing power values as you increase the frequency of the notes within the chords observed (scaled such that the highest power value is 1), the root within the data always contains the highest power value. A summary of the note power values (average and standard deviation) can be seen in the table below:

TABLE 8 - SINE WAVE POWER SUMMARY

Chord Type	Root	Std.Dev	2^{nd} Note	Std.Dev	3^{rd} Note	Std.Dev
Major	1	0	0.793667	0.017238	0.666833	0.017808
Minor	1	0	0.842833	0.017454	0.670417	0.013865
Augmented triad	1	0	0.794417	0.012665	0.63	0.01
Diminished triad	1	0	0.8435	0.01607	0.709833	0.01482

Here the low standard deviation for all notes in the series for all chord classes suggests that this instrument acts as an acceptable control instrument, as it has extremely predictable and reliable data values. The reasons however for the decrease in power values as you increase the frequency of a note could be numerous. A more detailed analysis is shown in later chapters. A sample of the tabulated results is shown in the table below. This, again, shows the predictable nature of the results:

MAJOR	C	C#	D	D#	E	F	F#	G	G#	A	A#	B
C	1.00				0.81			0.67				
C#		1.00				0.76			0.67			
D			1.00				0.79			0.66		
D#				1.00				0.78			0.65	
E					1.00				0.81			0.66
F	0.72					1.00				0.82		
F#		0.67					1.00				0.77	
G			0.67					1.00				0.79

FIGURE 9 - SAMPLE DATA SET FOR GENERATED SINE WAVE CHORDS

Given the data set found above, it should be trivial to transcribe the results from this into chords. A detailed analysis of the transcription rate is found in chapter 6, along with a discussion of the trends and meaning of the data obtained.

5.3 Synthetic Piano Chords

The following results were slightly more varied than the sinusoidal generated chords. This is expected as additional frequencies are present within the audio sample due to instrumental overtones, or, more simply, the ‘instrumentation noise’. A typical chromagram obtained from the analysis can be found in the figure below.

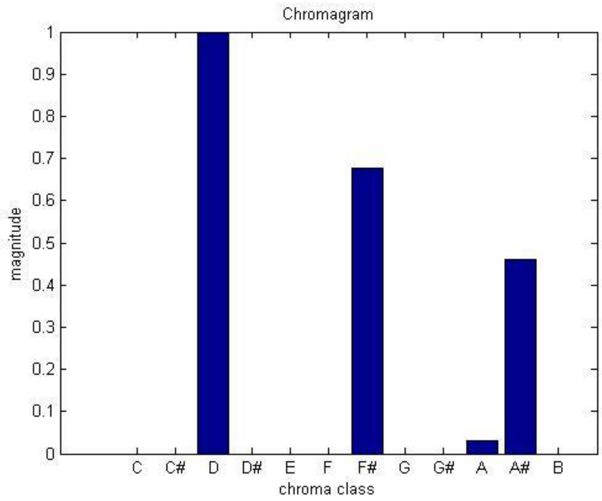

FIGURE 10 - SYNTHETIC PIANO D AUGMENTED CHORD

Here it is apparent how additional frequencies are present in some data samples, where the additions of the 'A' note within the chord (according to theory, this should not be included) shows this effect. When applying a Fast Fourier Transform to the D Major chord, a clearer picture can be observed.

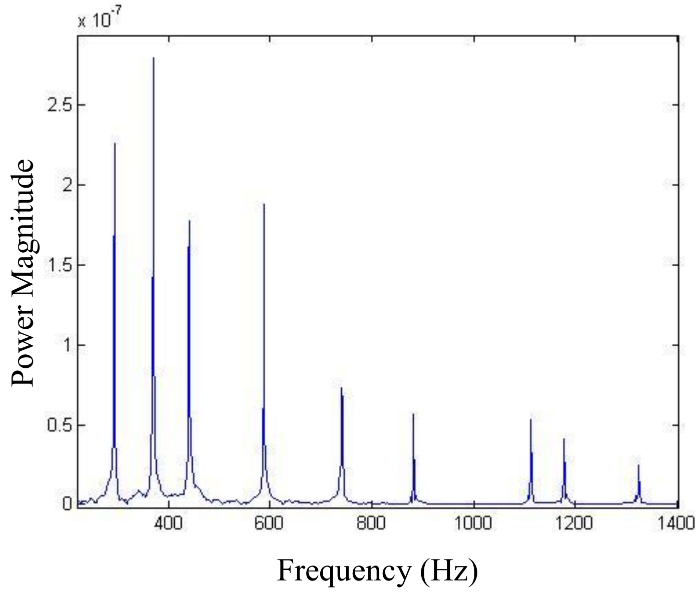

FIGURE 11 - FREQUENCY ANALYSIS OF A PIANO D-AUGMENTED CHORD

Where the overtones are highlighted in the following diagram:

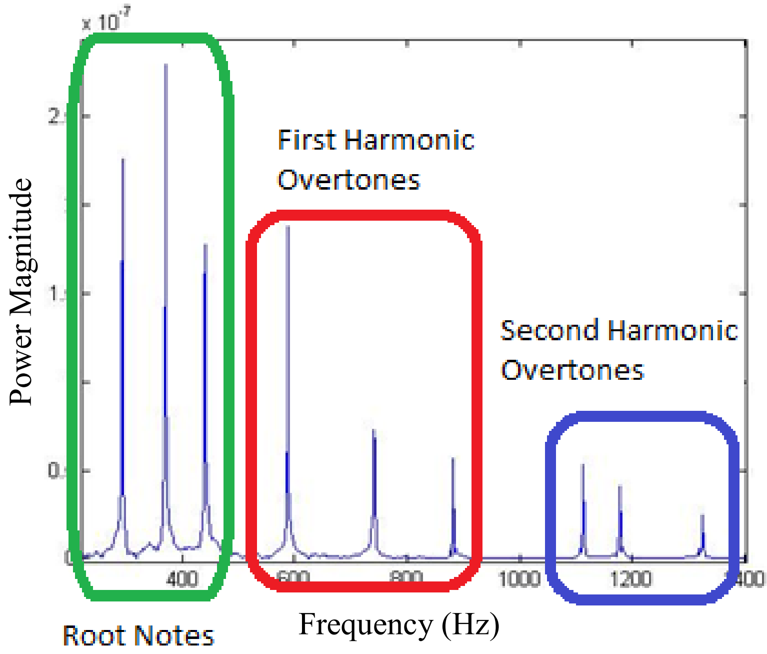

FIGURE 12 - SPECTRAL CONTENT SHOWING OVERTONES OF A D AUGMENTED CHORD

The first three peaks apply to the root frequencies of the notes [D, F#, A#]. The peaks occurring after the three first peaks are those of the instruments overtones, and can alter the data to some degree. Since the 'mirchromagram.m' function uses spectral wrapping as part of its process, the first harmonic overtones will simply be at an octave of the original notes. This means that they will just contribute more power to the related note value within the chromagram and only alter the power content of the final transcribed data. The third, and successive, harmonic overtones however, are at a frequency of 3 times the root. This is not considered the same note value (due to the exponential relationships between notes), meaning that additional notes not relating to theory could be introduced into the data set. However, as observed in the figure above, this often does not introduce high power content and a simple threshold introduced into the chromagram function can reduce or remove the effects of overtones easily. This is utilised in the 'mirchromagram.m' function and, as long as this factor is not changed throughout the experiment, an accurate comparison in terms of the scope of this thesis can be achieved.

A sample of this data is found in the table below (full results can be found in appendix chapter 10.3).

MAJOR	C	C#	D	D#	E	F	F#	G	G#	A	A#	B
C	0.72				1.00			0.65				0.02
C#		0.87				1.00			0.88			
D			1.00				0.39			0.11		
D#				1.00				0.62			0.47	
E					1.00				0.54			0.48
F	0.67					1.00				0.86		

FIGURE 13 - SAMPLE SYNTHETIC PIANO CHORD DATA SET

Analysing the average data trend gives the following table.

TABLE 9 - AVERAGE DATA TRENDS OF SYNTEHTIC PIANO CHORDS

Chord Type	Root	Std.Dev	2nd Note	Std.Dev	3rd Note	Std.Dev
Major	0.9656	0.0824	0.7235	0.1647	0.4972	0.1817
Minor	0.9345	0.1467	0.7725	0.1451	0.4700	0.1467
Augmented triad	0.9636	0.0851	0.7183	0.1589	0.412	0.1699
Diminished triad	0.9300	0.1575	0.775	0.1437	0.5212	0.1344

It can be observed that the standard deviation for each note is roughly 10 times that of the generated sine wave chords. Since human errors were removed by producing the sound via a musical synthesiser, this variability is solely due to the instrumentation noise of the piano itself. A more detailed discussion of the implications of this is found in chapter 6.

5.4 Electric Guitar Chords

After introducing the human element into the data, such that this experimental set contains instrumentation and human errors, a typical chromagram obtained from the data set is found below.

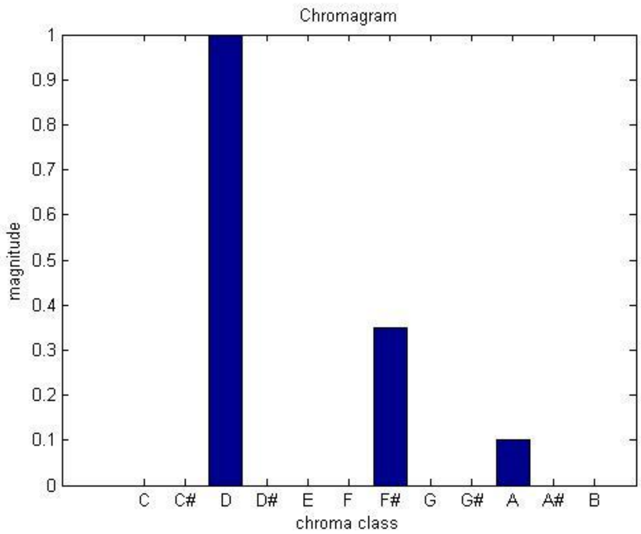

FIGURE 14 - ELECTRIC GUITAR D MAJOR CHORD

Applying a Fast Fourier transform on this chord shows the following diagram.

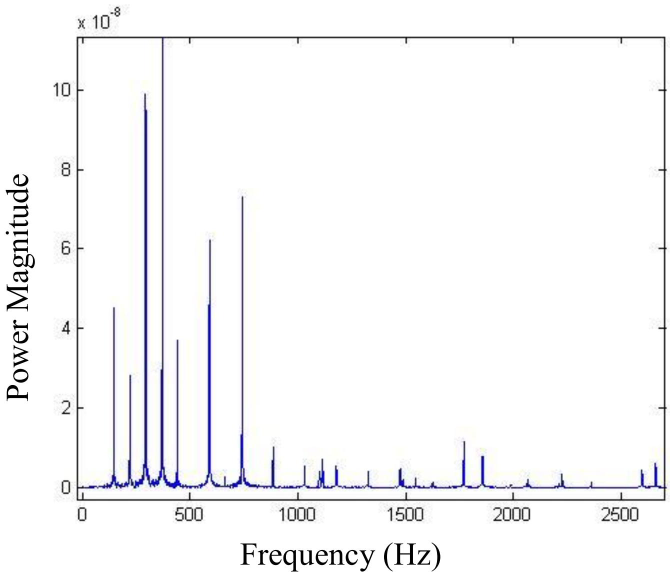

FIGURE 15 - D MAJOR CHORD SPECTRUM FOR ELECTRIC GUITAR

Two effects on the frequency content can be seen that differs from the control. The first effect is the inclusion of musical overtones which was previously discussed. The second, however, is the method of which chords are played. On a guitar, chords are played using the triad chords based on note content but notes each within the 3 note set can be played multiple times on multiple strings such that two or more notes at octave equivalence are present in the sample. This is evident as only 3 different notes are observed in the chromagram but a clustering of 5 frequencies in the root note spectrum (before overtones are introduced). This could introduce the error of largely varying note power value within the chromagram since octave note powers are simply added in the chromagram, and therefore other notes would be scaled accordingly.

The human element is also clearly observed by noticing the variable note power value for the original 5 peaks (which are the root notes actually played). Comparing this with the piano frequency transform shows how a human playing the stringed instrument can add large variations into the frequency and power content. This could potentially affect transcription accuracy.

A sample data set is also observed which shows how variable the data can be.

MAJOR	C	C#	D	D#	E	F	F#	G	G#	A	A#	B
C	1.00				0.70			0.21				
C#		1.00				0.41			0.07			
D			1.00				0.35			0.10		
D#				1.00				0.69			0.61	
E					0.72				0.07			1.00
F	1.00					0.66				0.17		
F#		1.00					0.46				0.14	
G			1.00					0.32				0.48

FIGURE 16 - SAMPLE ELECTRIC GUITAR SAMPLE SET

An analysis of the trend of the data across all 4 chord classes is shown in the following table.

TABLE 10 - DATA TREND OF SYNTHETIC PIANO CHORDS

Chord Type	Root	Std.Dev	2nd Note	Std.Dev	3rd Note	Std.Dev
Major	0.822417	0.229165	0.433067	0.212191	0.726817	0.36589
Minor	0.8175	0.231177	0.43295	0.26986	0.70	0.328386
Augmented triad	0.76	0.282803	0.555742	0.316336	0.58575	0.321604
Diminished triad	0.754325	0.244663	0.639522	0.294003	0.591083	0.266085

Several major differences in the data are observed when comparing to the last two data sets, especially in the Minor chord trend. Previously, the root note showed on average the highest power with a decrease of the higher notes in the scale. Here the 3rd note, and therefore the highest fundamental frequency, has a power value similar to the root value. Also note the standard deviation being roughly 3 times larger than that found in the synthetic piano set and 30 times larger to the control.

This suggests that assuming there will be some form of predictability in the chord profile obtained will cause errors to occur in the final transcription product. Although this assumption would simplify the process considerably, it is important that not assuming this fact and adjusting the process is done to achieve maximum efficiency of the process.

5.5 DISTORTED ELECTRIC GUITAR CHORDS

In this sub-experiment, only 3 chords were recorded and analysed to simply confirm the suspected effect that heavy distortion would have on an electric guitar's signal spectral content. The following chromagram was observed when analysing a 'Power chord':

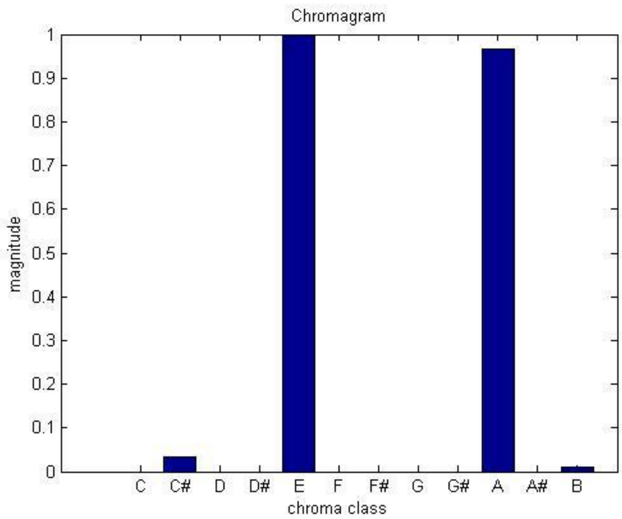

FIGURE 17 - DISTORTED GUITAR A POWER CHORD

Observing the frequency and power spectrum gives the following graph.

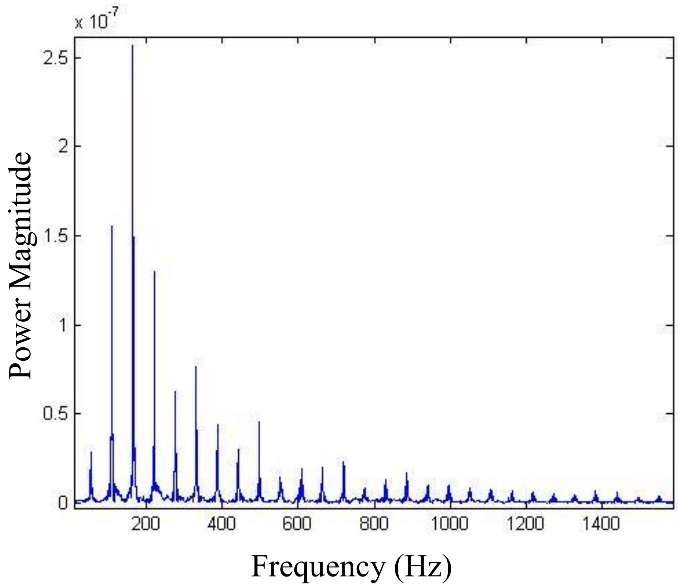

FIGURE 18 - DISTORTED POWER CHORD A SPECTRUM

The spectrum shows a periodic pattern in the peaks, where a large amount of harmonics are present due to the high distortion. This does add some additional noise content, but still gives a relatively clear chromagram which is the most important aspect when considering automatic transcription efficiency.

The full list of results is shown below. These results clearly outline the close correlation between the theory and practical data. The first table contains the theoretical values and the second table contains the obtained data.

Theory	C	C#	D	D#	E	F	F#	G	G#	A	A#	B
Power chord E					1.00							1.00
Power chord G			1.00					1.00				
Power chord A					1.00					1.00		

FIGURE 19 - DISTORTED GUITAR SAMPLE DATA SET THEORY

The practical results from experiments show the following (here outliers with value less than 0.05 relative power content were removed for clarity).

Data	C	C#	D	D#	E	F	F#	G	G#	A	A#	B
Power chord E					0.99							1.00
Power chord G			1.00					0.99				
Power chord A					1.00					0.97		

FIGURE 20 - DISTORTED GUITAR SAMPLE DATA SET FROM SAMPLE

A short discussion on the implications of these results can be found in chapter 6.4.

5.6 Acoustic Guitar Chords

This experiment yielded similar results to the electric guitar but with some extra noise introduced from the recording techniques needed to capture the acoustic guitars sound. A sample of a chromagram can be seen in the figure below.

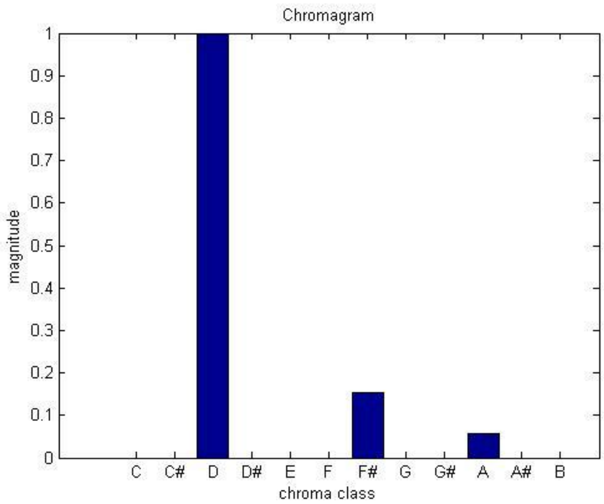

FIGURE 21 - D MAJOR ACOUSTIC CHORD CHROMAGRAM

It is noted that the power values observed for a chord can vary a large amount across the data set due to the combinations of noise. Comparing the chromagram for the D-major chord performed on the electric guitar, it is observed that the two additional notes played do not contribute the same relative power level (within a degree of accuracy). The frequency response also shows some of the noise content.

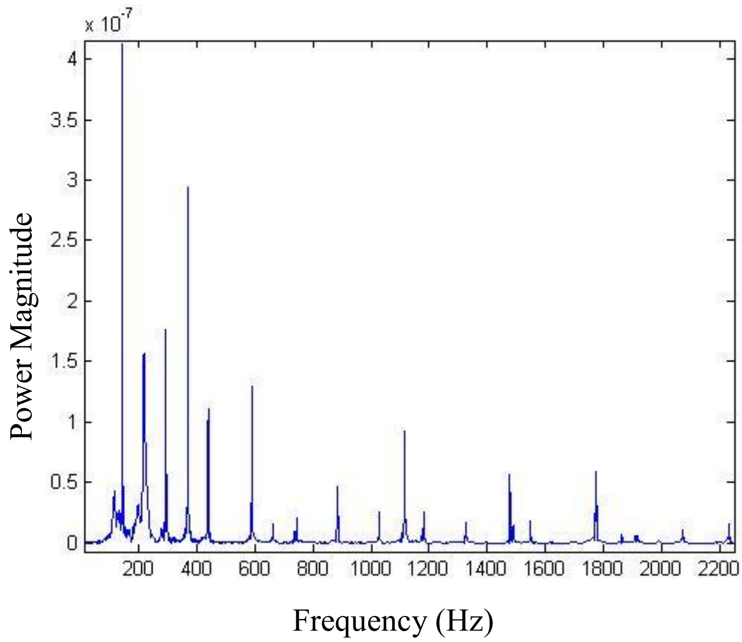

FIGURE 22 - D MAJOR ACOUSTIC FREQUENCY SPECTRUM

The ‘flaring’ seen at the base of the peaks in the lower spectrum suggests more noise when compared to the electric guitar samples. Since a highly sensitive sound recording device was used for this (standard method in professional studios), sound proofing was used to reduce any background noise in the audio samples. Regardless of the steps taken, unwanted noise is still present.

A sample of the data obtained is seen in the table below.

MAJOR	C	C#	D	D#	E	F	F#	G	G#	A	A#	B
C	1.00				0.48			0.19				
C#		1.00				0.56			0.08			
D			1.00				0.15			0.06		
D#				0.46				1.00			0.54	
E					0.56				0.09			1.00
F	1.00					0.33						
F#		1.00					0.31					

FIGURE 23 - SAMPLE DATA SET FOR ACOUSTIC GUITAR CHORDS

Observing the data trend across all 4 chord classes gives the following table.

TABLE 11 - DATA TREND OF ACOUSTIC GUITAR CHORDS

Chord Type	Root	Std.Dev	2nd Note	Std.Dev	3rd Note	Std.Dev
Major	0.744333	0.27736	0.404725	0.3663	0.652917	0.362922
Minor	0.69425	0.33393	0.310167	0.347301	0.66	0.315588
Augmented triad	0.790917	0.1527	0.426167	0.305719	0.438833	0.302367
Diminished triad	0.36225	0.359398	0.388833	0.371258	0.353833	0.332129

The most important thing to note from these results is that firstly, some chords (when played in the same way as the electric guitar) omit some notes from the chromagram altogether. This is most obvious in the Diminished data set where 30% of the theoretical notes were not present in the collected data set (11 notes across the 12 chords in total).

Also from the average data trends, a larger variability in the notes can be found. A large standard deviation was seen across the whole data trend, where in some cases the standard deviation matched the average power value (giving a standard deviation of almost 100%). From the spectral content seen in figure 16 above, it is obvious how dominant the bass frequencies are compared to previous results. This may be the reason for the omission of some notes. More details on this phenomenon are found in chapter 6.

5.7 FULL PRODUCTION CHORD SAMPLES

There are two sub-categories to this experiment. The first is without post-production effects and the second is with such musical effects which could potentially alter the data. The song was recorded as per the methodology set out at the start of the experimental work. The song was written to try and emulate real-world song's musical qualities. These exact qualities will not be included in this thesis due to the wide variability and non-systematic method of producing a 'standard song' (also realising there is no such thing as a 'standard song'). This experiment was performed to detail the effects of additional musical qualities such as percussion, bass and vocals. The purpose of the second part of the experiment is to see how audio effects alter these results.

The song was written in the key of C Major, where the following chord progression was used:

Verse	**Chorus**
C-Major, E-Minor, A-Minor, F-Major	C-Major, G-Major, F-Major, G-Major

5.7.1 No-Effect Full Production Song

After recording a song and then separating each chord manually so that individual analysis can be performed, a sample of a typical chromagram is provided below.

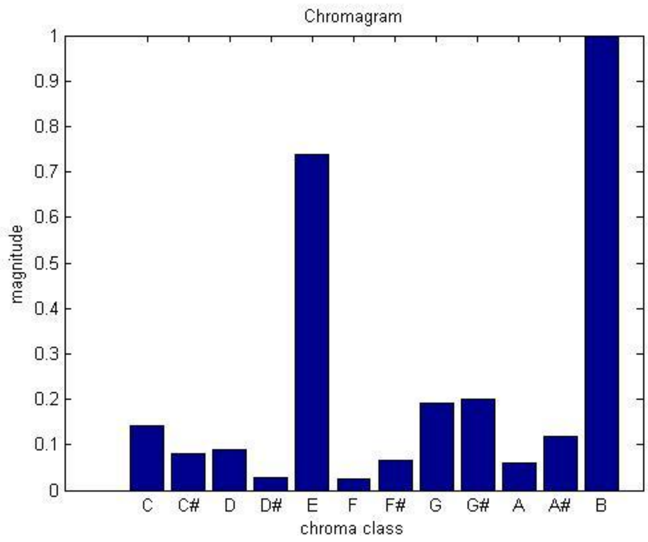

FIGURE 24-NO EFFECT E-MINOR CHORD WITHIN FULL PRODUCTION SONG

The most obvious observation that can be made is the fact that every note class has at least some spectral content in the chromagram. This is most likely a result of the wide-spectrum percussion sounds adding additional 'noise' (where noise in this context is any notes that do not relate to the underlying chord being played). The peaks of the graph above still show dominance in the desired note-values to allow accurate transcription via the 'highest likelihood' comparative technique.

Having a look at the spectral content via a fast Fourier transform gives the following graph.

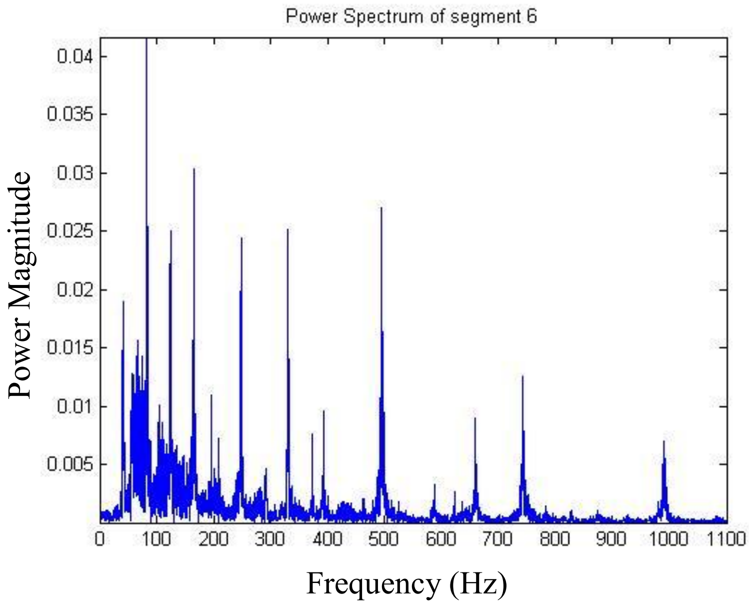

FIGURE 25 - FREQUENCY TRANSFORM OF FULL PRODUCTION SONG SEGMENT 6

The additional noise can be clearly observed but with several distinct peaks being shown. The highest value should also be noted to be in the low frequency spectrum (0-100Hz) suggesting it is from the bass instrument.

Tabulating the data does require some modification to the methods used previously, with several columns and rows being added to display all relevant information. A sample of the collected data is found below.

Segment	Percussion	Electric Guitar	Bass	Vocal	Chord	C	C#	D	D#	E	F	F#	G	G#	A	A#	B
1	x	x			C Major	1.00				0.33			0.19	0.06			0.02
2	x	x			E Minor	0.08				0.79		0.06	0.09	0.06			1.00
3	x	x			A Minor	1.00	0.05	0.07	0.04	1.00			0.14	0.14	1.00		0.04
4	x	x			F Major	1.00	0.03				0.69		0.02	0.04	0.19		
5	x	x	x		C Major	1.00				0.25			0.10	0.08			
6	x	x	x		E Minor	0.14	0.08	0.09		0.74		0.07	0.19	0.20		0.12	1.00

FIGURE 26 - SAMPLE DATA SET FROM FULL PRODUCTION SONG NO EFFECTS

The bass note followed the root of the chord, which is commonly the case (it can and often does differ from the root but will still be included in the underlying chord being played). Transcription accuracy will be observed in the next chapter.

5.7.2 EFFECT-ADDED FULL PRODUCTION SONG

After adding effects to the individual tracks above, the same analysis was performed on the same segments. A quick summary on the effects applied to each track is given below.

Track	Effect 1	Effect 1 value	Effect 2	Effect 2 value
Percussion	Compression	10% max volume		
Electric guitar	Compression	10% max volume	Reverberation	15% raw mix
Bass	Compression	10% max volume		
Vocals	Compression	10% max volume	Reverberation	10% raw mix

FIGURE 27 - DETAILS OF EFFECTS SETTING USED IN FULL PRODUCTION SONG WITH EFFECTS

For the compression mix, 10% maximum volume means that a compression is applied to the mix if the volume is below or above 10% maximum volume of the track (where 101% full volume is where clipping occurs). The amount the volume is adjusted is a dynamic and complex process and is not required to be fully known in this context. The reverberation effect assumes that the reverb effect is completed in two stages. The first stage applies the reverb effect to alter the sound according to the simulated reverb environment. The second stage then mixes this sound in to the original sound. 15% raw mix means that the reverb alters the sound and then adds it to the original track at 15% reverb sound and 85% unaltered sound.10% reverb was chosen for the vocals where these values were used based on several years of personal experience in music production and recording.

The same chord as the no-effect song is analysed using the 'chromagram' spectrum with the following output.

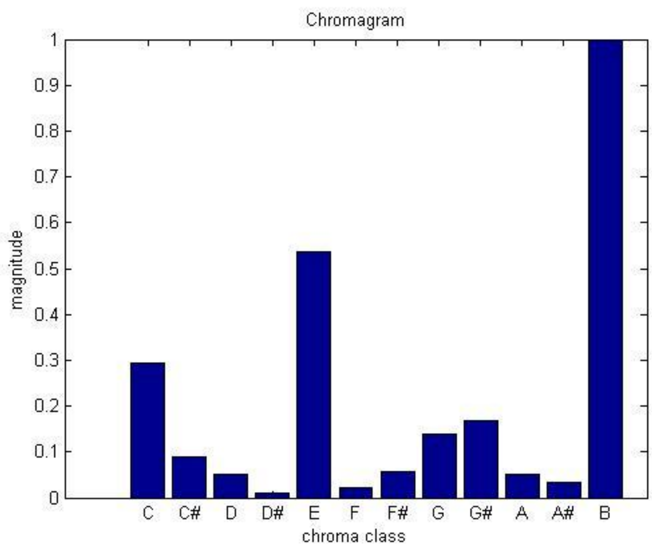

FIGURE 28 - CHROMAGRAM FOR FULL PRODUCTION SONG WITH EFFECTS E MINOR CHORDS

A few changes can be observed in comparison to the previous sample, where the major difference is slight power variations from the previous results.

The spectral content also shows some variation.

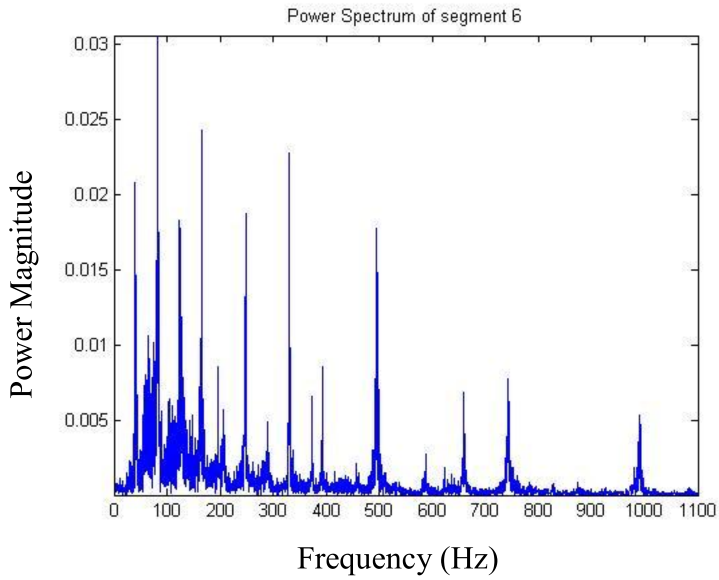

FIGURE 29 - FREQUENCY TRANSFORM OF SEGMENT 6 FULL EFFECTS SONG

The compression effect can be immediately noted by the maximum peak having a power of approximately 25% less than the previous results. A sample data set is shown below.

Segment	Percussion	Electric Guitar	Bass	Vocal	Chord	C	C#	D	D#	E	F	F#	G	G#	A	A#	B
1	x	x			C Major	1.00				0.35			0.18				0.01
2	x	x			E Minor	0.11				0.77		0.08	0.12	0.07			1.00
3	x	x			A Minor	0.80				0.63				0.08	1.00		
4	x	x			F Major	1.00					0.74		0.04		0.19		
5	x	x	x		C Major	1.00				0.22			0.23	0.05			
6	x	x	x		E Minor	0.30	0.09	0.05		0.54			0.14	0.17			1.00
7	x	x	x		A Minor	0.24				0.66				0.26	1.00		

FIGURE 30 - SAMPLE DATA SET OF FULL EFFECTS SONG

The next section will compare the results based on the effect the noise content has on transcription accuracy. This is the most important property of the automatic transcription property.

5.8 Conclusion

After recordings and analysing several different instruments in isolation, along with a final production song, several key issues were identified within the music. The design of the experiments performed has given enough data to study the trends of introducing noise into musical audio. This means it can be deduced the source and effect of these noise elements, and how they can be reduced to improve the overall accuracy of music transcription.

The large amount of data collected from several different instruments under different conditions highlights the key aspects of noise within the audio. This can be further analysed in the context of how it relates to the actual transcription of the data, this will be the focus of the next chapter. It is understood that even though it is impossible to design an experiment that will encapsulate the variability of data in the real world, the results here provide a good overview on the possible noise elements that are currently contained within music. The data collected in these experiments therefore should not be considered as an accurate description of what a final piece of music will look like, but rather a means to accurately and consistently correlate the effects of noise between samples.

The use of a control (the generated sinusoidal chords) allowed for future experiments that use more music-central methods of recording sound to be compared with a degree high enough to draw accurate conclusions about the noise within music. This chapter focused on identifying if noise is present within the audio, the next chapter will focus on why it is present, and a method of reducing its effect. This will then lead to the final chapter which gives suggestions to the current automatic transcription of music such that the accuracy can be increased (where the inaccuracy is due to the noise sources identified).

6 FINDINGS AND DISCUSSION

To correctly identify the noise sources within data which an automatic transcription process will be applied to, each experimental setup is again analysed in isolation to allow for not only what effect a specific noise source has on the audio data, the extent of its effect, but also how this might affect the automatic transcription process. Using a 'highest likelihood probability' algorithm, each data set is compared to the binary data set corresponding to the theory. Applying the basic transcription method for each of the isolated chord data sets that were gathered, a few outputs will be given, including the transcription accuracy, correlation with the theoretical chords and which chord class produced the best results. Discussing each result set in order in which errors are introduced, first an analysis of the control set (the generated sine wave chords) will be conducted.

6.1 SINE WAVE GENERATED CHORDS

With all the data collected, the first and most obvious observation as seen in the chromagram in figure 5 is the exponential decrease in note power values as the frequency is increased. This behaviour is observed in all the subsequent data obtained which suggests it is a global effect and not isolated to the generated waveforms. To identify exactly what relationship this decrease is (to aid in the identification of this phenomenon), the data points for the Major chords were graphed and a line of best fit was applied. The following table summarises the results and trends relating to the data obtained.

TABLE 12- EQUATIONS OF BEST FIT FOR THE DATA POINTS GENERATED SINE CHORDS

Chord Type	Line Of Fit	Equation	R^2
MAJOR	Linear	$-0.1666x + 1.1533$	0.9712
	Exponential	$1.2132e^{-0.203x}$	0.9801
	Power	$1.0059x^{-0.365}$	0.9819
MINOR	Linear	$-0.1648x + 1.1673$	0.9902
	Exponential	$1.2332e^{-0.2}$	0.9827
	Power	$1.0197x^{-0.352}$	0.9367
ALL COMBINED	Linear	$-0.1657x + 1.1604$	0.9633
	Exponential	$1.224\mathrm{e}^{-0.202\mathrm{x}}$	0.9569
	Power	$1.013x^{-0.359}$	0.9347

These results show that this decrease is in fact not randomly produced and therefore is superimposed onto the data from, most likely, a single source within the process from generation to analysis. The suspected sources for this are either the sound generator (which was developed for musical use) imparting this behaviour into the sound, or an implication brought upon via using a fast Fourier transform. It could also simply be attributed to the three sine waves interacting together thus resulting in higher frequency sounds not containing as much power. Investigating this phenomenon requires several steps which focus on the software used to analyse the sound file.

To observe if it is the analysis software or the generated sound, a simple MATLAB function was written to construct a wave containing three sine waves at musical frequencies. The program simply generated three sinusoidal waves in a vector at the same sampling frequency as the audio used throughout the experiment. These three waves were then simply added and then exported into a .WAV file using the inbuilt MATLAB 'wavwrite.m' function (the full function can be found in the appendix chapter 10.8). This audio file (which was confirmed to be the same as the generated sound file by listening to the sound) was then analysed using the 'mirchromagram.m' function yielding the following results.

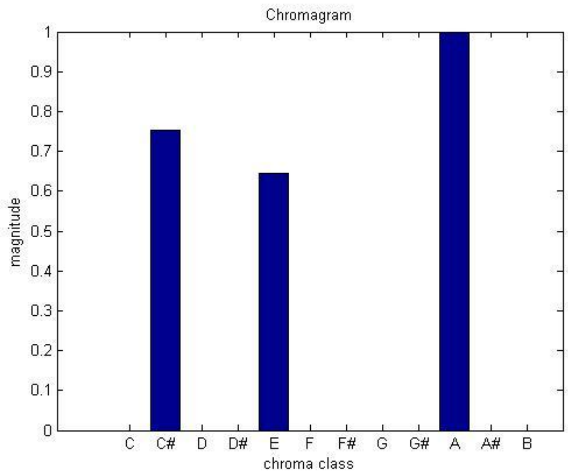

FIGURE 31- CHROMAGRAM OF A TYPICAL CHORD SHOWING DECREASE IN POWER

This means that this error was not introduced by the generated sound, but most likely within the 'mirspectrum.m' function used within the chromagram process. A look at the spectrum from this function shows the following.

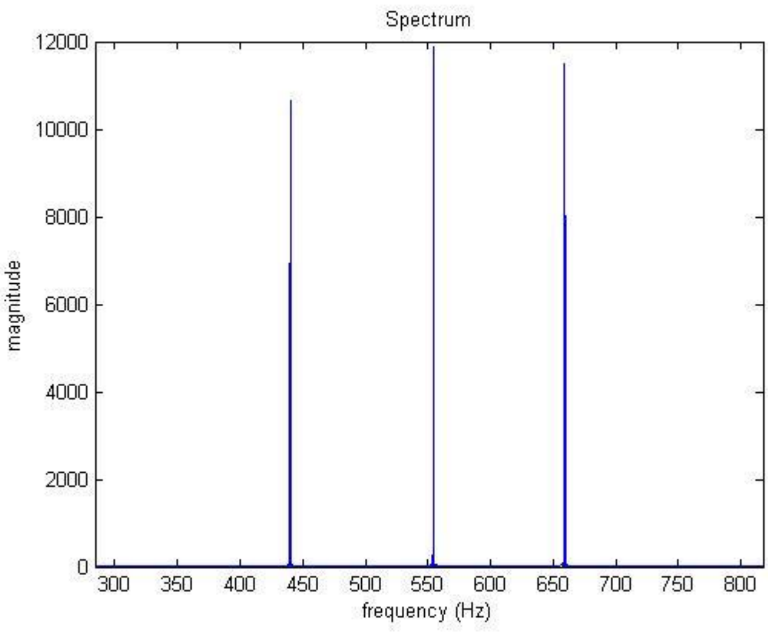

FIGURE 32 - FREQUENCY TRASNFORM OF SINUSOIDAL GENERATED CHORDS

Observing the program itself showed many variables set within the function and the ones that relate to the fast Fourier transform were identified and altered. The spectrum found here is a result of increasing the resolution of the transform calculations by a factor of 16:

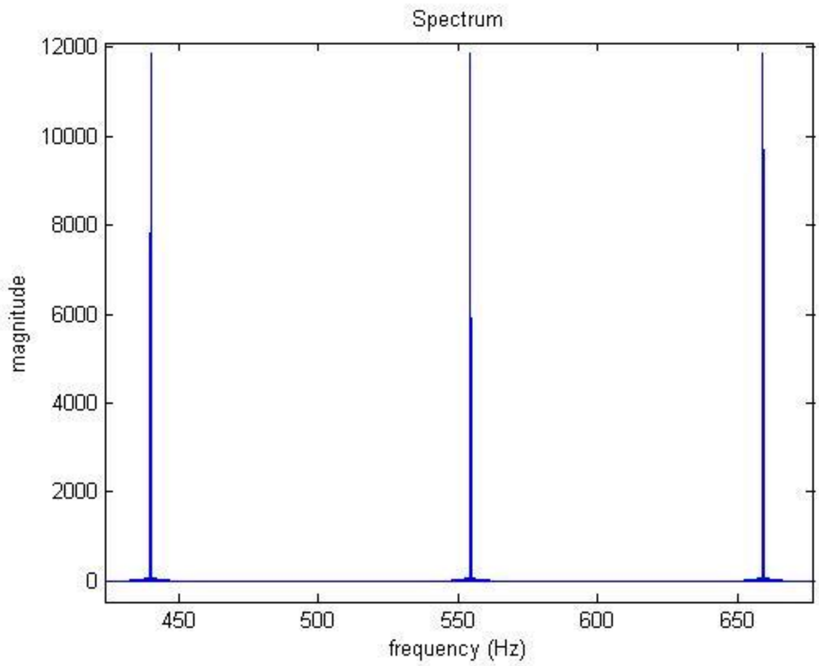

FIGURE 33 - IMPROVED VARIABLES FREQUENCY TRANSFORM

This figure shows a much more accurate transform. Here 16 was chosen as the multiplier, as this was the lowest power of 2 (a required parameter for FFT) where a true representation of the waveform was seen. This did however increase the computation time from 0.188 seconds to 0.776 seconds, an increase of 312%. This time was calculated using a high performance computer and therefore would only increase when lowering the ‘power’ of the machine used to process the audio. This means there is a trade-off between speed and accuracy. Given this sacrifice in time only yield an 8% increase in power value accuracy of the highest error sine-wave, this is not a reasonable trade-off.

Altering the necessary parameters in the ‘mirchromagram.m’ function also increases the computational time by roughly 800%, where along with the 312% increase in the spectrum analyser, shows that the variables used within the functions used in the analysis of this thesis have been optimised for ‘speed vs. accuracy’. Also considering the implication of a real-time use of the complete final process, this is a vital trade-off that needs to be considered in future projects.

When applying the maximum likelihood algorithm to the data obtained in this experiment, a 100% transcription rate was obtained for all of the 48 chords. This shows that a reliable transcription method can still be performed using the non-optimised (relating to accuracy) function. The following table summarises the transcription outputs.

TABLE 13 - TRANSCRIPTION ACCURACY OF GENERATED SINE CHORDS

Chord Type	Accuracy	Average correlation
Major	100%	2.2492
Minor	100%	2.3058
Augmented	100%	2.4217
Diminished	100%	2.3442

This gives a baseline for the following analysis of alternative instrumentation and recording parameters. An average correlation of approximately 2.3 was observed across all chords, this being due to the exponential decrease in power values discovered earlier. This control experiment not only validates the transcription method used here, but also that a perfect accuracy can be achieved when all the noise is minimised. Next we move onto the synthetic piano which contains instrumental noise.

6.2 SYNTHETIC PIANO CHORDS

In this experiment, the effect of instrumental noise was observed due to the implications of sound waves created from a vibrating string. However, the implication of this should have minimal effect on the data to allow for further accurate development of an automatic transcription process. As observed in chapter 5.3, a frequency analysis of the sound file shows the additional frequencies due to upper partials. A few additional analyses can be performed to justify that these do not interfere with the transcription process.

The same C-major chord was used from the audio samples as the control as it showed the most additional frequencies in the spectrum. First a rough estimate of the power of these upper partials (which occur at integer multiples of the fundamental frequency, which would not necessarily correspond to the same 'note' value) was done by observation and then a similar function was written similar to the Sine wave analysis section to investigate the fast Fourier transform. In this case however, additional upper partials were also added into the final signal to apply the chromagram function to. The difference between these two will show the effect of these upper partials. The following addition to the code used previously was made (see appendix chapter 10.8 for code used). This code will generate the sine waves to emulate the effect of upper partials. However, the analysis will first be done using the fundamental frequencies only. The following spectrum was found.

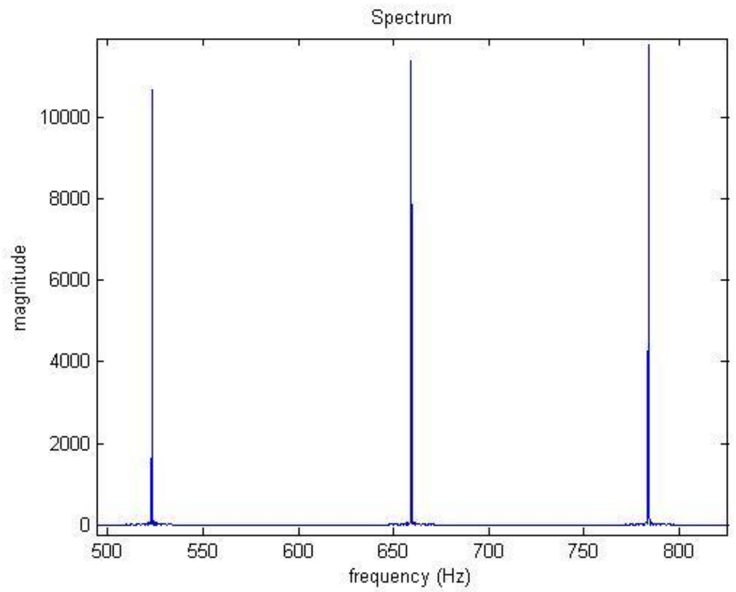

FIGURE 34- BEFORE ADDING SYNTHETIC UPPER PARTIALS

This is the expected result. When adding the additional frequencies, the following spectrum was found.

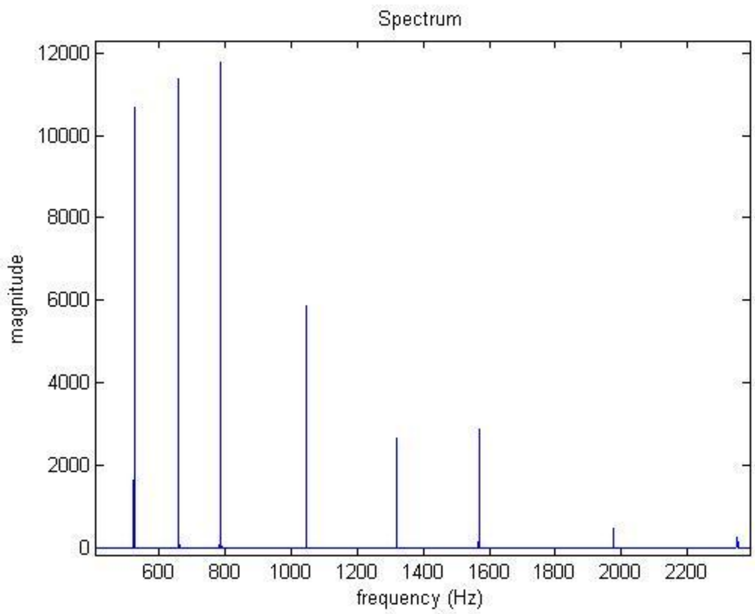

FIGURE 35 - AFTER ADDING UPPER PARTIALS

When compared to the equivalent spectrum found from the piano chords generated with real piano samples, the following spectrum was obtained.

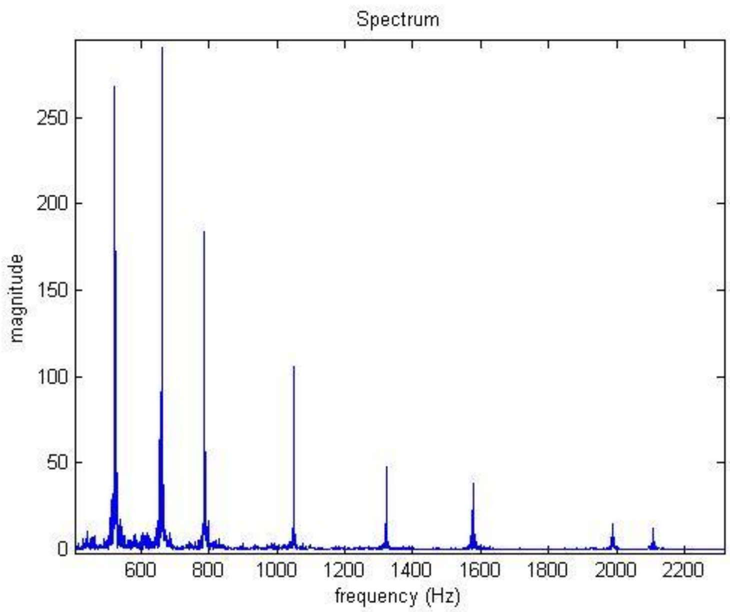

FIGURE 36 - FREQEUNCY TRANSFORM OF ACTUAL PIANO CHORD

This shows comparable features (where the real piano recording contained more noise, as expected when recording the instrument samples using a microphone). When comparing these results in terms of the chromagram data that can be extracted, the following table is constructed.

TABLE 14 - EFFECTS OFUPPER PARTIAL ON CORRELATION

Sample type	C	E	G	correlation
No upper partials	1	0.775	0.63	2.405
With upper partials	1	0.675	0.548	2.223
Real piano sample	1	0.721	0.650	2.371

In this table, the three audio samples are listed followed by the three notes that are contained within them (the fundamental notes for the cases where upper partials are present). The power values are recorded in the appropriate entries and then finally the correlation value used by the highest likelihood algorithm is shown to observe the strength of the transcription.

The first thing to notice is that adding the upper partials into the sound does decrease the correlation with theoretical binary values by around 7%. This could cause a problem in the automatic transcription in some rare cases where the difference between correlation of a correct and incorrect chord is less than this. Thus causing incorrect transcription, but this is an unlikely case and therefore this effect is seen to be present but insignificant at this time. Since this is purely a noise source that is added to the sound at the time of production, there is nothing that can be done to reduce this besides frequency filtering. This however is not a viable option in the case where other musical features or instruments are contained within the audio at these upper frequencies.

When actually transcribing the data obtained from the synthetic piano chord samples, the following table is constructed.

TABLE 15 - TRANSCRIPTION ACCURACY OF SYNTHETIC PIANO CHORDS

Chord Type	Accuracy	Average correlation
Major	100%	1.9750
Minor	100%	1.9750
Augmented	100%	2.0950
Diminished	100%	2.0158

When comparing these values to the ones found in the sine wave data set, a lower average correlation is seen across all chord types. A transcription rate of 100% is still seen, but if the correlation average drops too low it would be more likely to produce an incorrect transcription due to other noise sources decreasing the difference between correlations of a chord set. To see if this is the case as more noise is introduced into the audio samples, we continue with the experimental analysis where the electric guitar is analysed next.

6.3 ELECTRIC GUITAR CHORDS

In this data set, the additional noise of the 'human element' is added into the sound files. Three main sources of error are expected from this data set. The first source is the method of playing chords which differs from the method of playing piano chords. Since the manner of which a guitar (or similar instruments) is constructed, six strings at tension are located side by side. There are thin metal bars or 'frets' located normally to these strings at intervals conforming to the logarithmic scale of frequencies. The six strings are then tuned to conform to the 'standard tuning' model which consists of the following notes in order of the highest frequency to the lowest: [E, B, G, D, A, E]. A player can use their fingers to press the strings against the metal frets, altering their wavelength and hence its frequency. This method of controlling the notes contained within the instrument means that up to six strings (or notes) can be played simultaneously. Often these contain only three different notes with one or more octaves of the same note being present in the audio sample. This implication, along with the observed behaviour that increasing the power content of a specific note can also decrease the other notes within the chromagram, can cause inaccurate transcriptions. This means that the data collected here would show more variability.

The second source is brought by the fact that strings need to be physically struck to produce sound, where the resulting sound waves amplitude is directly related to how hard the string is struck. Since the exact same amplitude cannot be imparted to all the sound waves, there will be large variability in the power content of each frequency within the chromagram which could reduce the transcription coefficient.

The final source is the analogue to digital (A2D) converter that is used to record the sound. However, when considering that this data is recorded at 48000 Hz which is actually at a higher sampling rate than the sound file (which is sampled at 44000 Hz), this should mean that no loss of information should occur at this stage of the signal chain.

The following scatter graph, showing the data distribution of each set, makes apparent the variable nature of the data.:

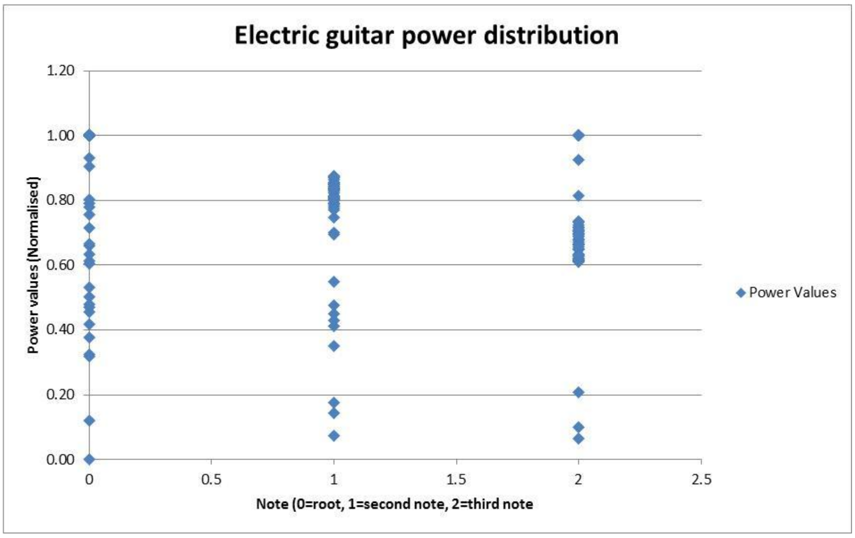

FIGURE 37 - ELECTRIC GUITAR DATA DISTRIBUTION

In the diagram above, the axis on the bottom represents the three notes contained within a chord, with the root note being the zero entry. This shows that for the root note, the power distribution is extremely varied, ranging at approximately an equal distribution from 0 to 1. The second and third note in the chords shows groupings at certain power values. Although it is not obvious from the graph, there is a large portion of root notes with a power value of 1. This means an equation of best fit can still be applied to the clusters, but since the data is so varied, this is no longer an accurate method of predicting power values as it was with the generated sine wave chords. This highlights the effect of human sourced noise and that predicting power values in a chord as a method of chord transcription is not a viable method to focus on.

To be able to successfully transcribe this data however, it can be observed that even though the distribution is more varied that previously, there are no outliers in terms of notes present in the chromagram that are not contained within the chord actually being played (as was the case with the synthetic piano chords). This can be attributed to the fact that playing multiple notes of the same value superimposes their power values which decreases the upper partials contributions when all the power values are scaled. This is the effect of a frequency (or three in this case) dominating the power spectrum which essentially filters out the other 'noisy' frequencies. This suggests that accurate transcription should still be possible.

However when observing the data set as a whole, there are a few chords where only two note values were recorded. A sample of this can be seen in the figure below.

AUGMENTED	C	C#	D	D#	E	F	F#	G	G#	A	A#	B
F#			1.00								0.81	
G				0.75				1.00				0.61
G#	1.00				1.00				0.12			
A		0.62				0.77				1.00		
A#			1.00								0.61	
B				0.31				0.15				1.00

FIGURE 38 - SAMPLE DATA TO OBSERVE EXCLUSION OF NTOES

This can be attributed to several reasons. Either it was not played in this specific chord (many variations of the same chord exist on a guitar) or that the other two frequencies dominated so much that the chromagram filtered it out. This could cause concern for chords within the Diminished or Augmented set, as the Major and Minor transcription would be favoured over these if the two notes actually present in the data sample are shared amongst two chord types. When actually transcribing the data set using the MLA, the following results were obtained.

TABLE 16 – TRANSCRIPTION ACCURACY OF ELECTRIC GUITAR CHORDS

Chord Type	Accuracy	Average correlation
Major	100%	1.8300
Minor	100%	1.8292
Augmented	100%	1.9025
Diminished	100%	1.8250

This is a positive result as 100% of the chords were correctly transcribed. It can be observed that the average correlation value is slightly lower than the equivalent results using the piano data set. This suggests a pattern where the more noise introduced into the audio sample, the lower this correlation value is, which is the main problem with incorrect transcription. Ideally, it is desired that the correlation with the correct chord to not only be high, but to be substantially higher when compared to other chords. Both of these effects have seen reduction as more noise is introduced, thus the analysis is continued to see if this trend is maintained.

6.4 DISTORTED GUITAR CHORDS

This experiment is an accompaniment, or sub-set, to the other data. The implications of analysing a distorted electric guitar sample are substantially different from other audio samples for two main reasons.

The first reason is the 'genre' implication. Generally genres where a heavily distorted guitar is used as its main instrument (such genres as heavy metal, punk, and various types of rock) do not use the full chord range which was used in the other experiments. This is due to the distortion effect where if you try and distort more than two sinusoidal waves simultaneously, the resulting sound is 'muddy' or 'dirty'. This translates into the inability for humans to perceive the three frequencies individually which a requirement to successfully identify the chords. As a result, the main types of chords used here are what are known as 'power chords', these only use the root note, the 7th note above the root and the octave of the root (12 notes above, or essentially the same note at double the frequency). This means only two different note values are included in each chord.

The second reason is the actual effects of distortion. The expected behaviour is that since the distortion amplifies a signal to be above the clipping point of the amplifier, all the clipped waveforms will contribute the same power level. When applying the chromagram function to a smaller set of data using a heavily distorted electric guitar, it is clear from the results that this is the case. An example of a chromagram is as follows.

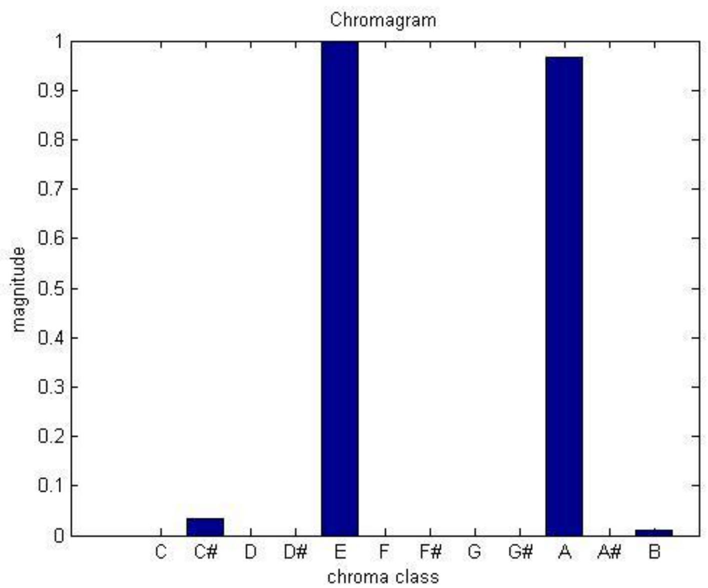

FIGURE 39 - DISTORTED GUITAR CHROMAGRAM

This clearly shows this effect to be an accurate assumption. This implication brings about an idea to be explored in the future development of this subject, wherein the information embedded in a song file (through the use of a digital data tag known as the 'ID3 tag') can be used to adjust certain parameters of automatic transcription process. This will be discussed in more detail in chapter 7 of this report. First we perform further investigation with noise associated with an acoustic guitar and a microphone.

6.5 ACOUSTIC GUITAR CHORDS

This experiment adds two more notable noise elements into the data samples, both of which add to the pre-existing noise elements found to be a factor in the previous experiments. These include the acoustic properties of an acoustic guitar and the recording technique used which utilises a studio microphone instead of direct input.

The method at which sound is amplified with an acoustic guitar differs from an electric guitar. As the names suggests, electric guitars use magnetic pickups to convert the motion of the metal strings into a useable signal. Acoustic guitars, however, use the body of the guitar, which is hollow and specifically designed to amplify the sound acoustically. This method of amplification can impart, and often does, its own sound qualities. Usually these are in the order of amplifying the low frequency spectrum more so than the high frequency spectrum. This effectively amplifies the feature where a dominant note in the chromagram reduces the normalised power values of other notes.

The second source is the use of a condenser microphone. This type of microphone is most commonly used when recording an acoustic guitar as it is considerably more sensitive than other microphones used with live musical performances. This could possibly have two effects to the final audio file, with the first being additional noise being picked up and the second is the microphone itself producing a filtering effect. Since the microphone would have internal capacitance and inductance etc. the final signal could be altered by a high-pass or low-pass filter. All these additional noise sources suggest that the final data values would be even more varied than the previous data sets and the graph below shows how the distribution of power values shows this:

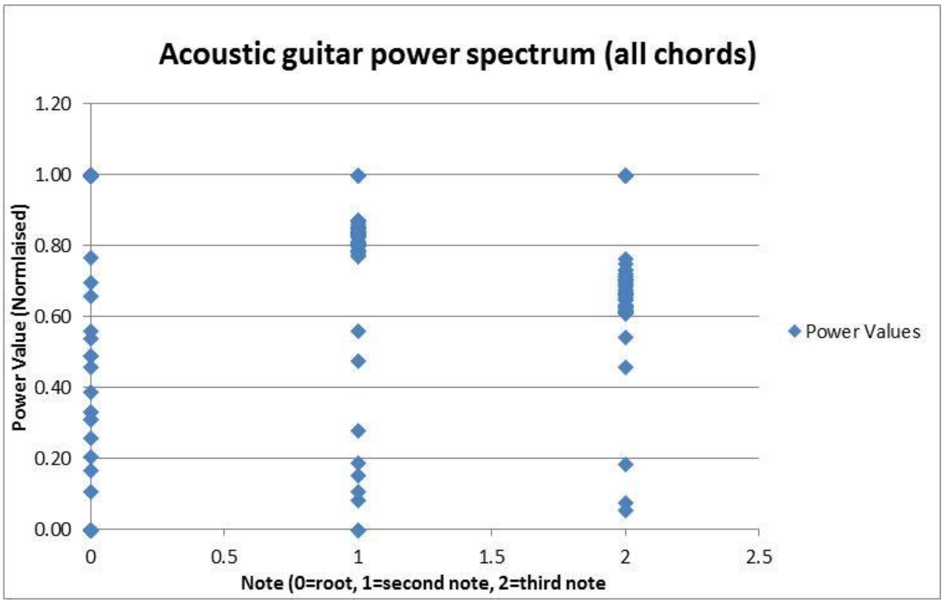

FIGURE 40 - ACOUSTIC GUITAR DATA DISTRIBUTION

As previously found, each successive note above has a cluster of values in a similar position as the previous experiments showed, but, in this case, much more variance is observed. Another effect observed in the data set is the exclusion of notes. This was also the case in the electric guitar data set but at a much more notable level.

A sample of this can be seen in the data set below.

AUGMENTED	C	C#	D	D#	E	F	F#	G	G#	A	A#	B
E	0.80				1.00							
F		0.42				1.00				0.11		
F#			1.00				0.12				0.92	
G				0.38				1.00				

FIGURE 41 - ACOUSTIC CHORDS DATA SAMPLE

When calculated, the whole data set of 48 chords showed that 18% of the chords contained a note missing from the theoretical values. When the transcription process was performed on the data, the following results were found.

TABLE 17 - TRANSCRIPTION ACCURACY FOR ACOUSTIC GUITAR CHORDS

Chord Type	Accuracy	Average correlation
Major	100%	1.5817
Minor	100%	1.5175
Augmented	58%	1.6558
Diminished	66%	1.6955

These results show the first inaccurate transcription. A few examples of these incorrect transcriptions are as follows.

TABLE 18 - INCORRECT TRANSCRIPTION ANALYSIS

Chord	Transcription	Correlation with actual chord	Correlation with transcription
C# Augmented	C# Major	1.36	1.36
E Augmented	E Major	1.80	1.80
D Diminished	F Minor	1.21	1.21

From these results, it is apparent that the incorrect transcriptions simply have their chord classes (major/augments/diminished etc.) incorrect. This is a result of favouring the Major and Minor chords over Augmented and Diminished in the cases where a note is omitted from the data. This means that the sample shows equal correlation between the two chord classes. In the case of the 'D Diminished' chord being transcribed to the 'F Minor' chords, this is due to the omitted note being the root note (where the D Diminished should contain [D, F, G#] but the data here only contained [F, G#]).This means that the chord 'F Minor' was a better match than the 'D Minor' chord.

Thus the reason for incorrect transcription between chord classes (still having the same root note description) can be contributed to the omission of a non-root note from the data. Similarly an incorrect transcription between chord classes can be attributed to where the root note was omitted. Both cases would be considered an error to the transcription, but it does show reasoning to use musical theory for error detection and correction. The following regards the case of a non-root note being omitted from the data set.

Music theory has proven that songs often are written in a single key, where this key dictates the types of chords being played. For example a song written in the key of ‘C’ will contain chords such as ‘C major’, ‘E Minor’, ‘F Major’, ‘G Major’, and ‘A Minor’ and others. So it should be theoretically possible to determine the key of a song if enough chords have been played, thus allowing the automatic transcription algorithm to use this information to correct any errors. If a chord is played where the data extracted contains the correct root note within the chromagram, but a non-root note is excluded, an incorrect chord class association will occur for that chord. If we take the augmented scale for example, if the root note and the second note are included in the data, but not the third note, the incorrect transcription will be a Major chord with the correct root note association. If the root and third notes are included but not the second, the transcribed chord will be a minor chord with a root note association five semi-tones above the actual root. Information like this (which would be different for varying situations) can then be used to correct the chord based on the key of the song and which notes are omitted from the data.

If a root note is omitted from the data, the correction techniques become slightly more complicated (in terms of what the possible transcriptions could be and how to correct this given the key of the song), however it should still be theoretically possible. It should be noted here that these error correction processes would take up additional computation time and further investigation should be done to see if the trade-offs between time and accuracy are justifiable to use if a real-time system is to be developed.

6.6 TIME SCALING

Thus far we have identified some of the potential errors that could occur in the type of data that would essentially be analysed with the automatic transcription software. However, up to this point, everything has been recorded and analysed in isolation. Although those experiments were important in identifying some root causes of error, it still does not fully encapsulate all the potential errors that could result in a completed piece of music. Parameters used in other aspects of the transcription process could also impart some error and more importantly the time scale at which you analyse the chromagram. In all the above experiments, equal time scales (roughly 4 seconds per chord) were used which means that it does not matter if the data is used for detecting the noise and difference between the sets. However in a real song, the time scale, used to analyse the chords over, could show differences in the chromagram.

The song written specifically for this part of the experiment used a simple chord progression in the key of C, where each bar of the song (a musical time scale which consists of 4 beats in this case per bar, where the time of each beat depends on the 'beats per minute' over which the song is played) contains a single chord. This was then separated manually and analysed in sections. This task of separating the chords can be done using other computational methods with a few functions available in MATLAB by the same author of the 'mirchromagram.m' function used in this thesis. As doing this task automatically was not part of the error investigation in this thesis, all segmentation was done manually.

Practically, the desired time scale that should be used when analysing the audio is the maximum time where each different chord is being played. This is to try and minimise the short time noise elements (such as percussion) or other notes that do not relate to the chord. Thus to minimise the contribution of these noise elements, different time scales of a chord will be analysed to see if this effect is the case.

Below, a table showing the results from such an analysis is shown. The chord being played here is 'A-Minor'.

Time scale	chord	C	C#	D	D#	E	F	F#	G	G#	A	A#	B	Transcription	Correlation	2nd Transcription	Correlation
Full bar	A minor	0.23				0.36			0.29	0.33	1.00			A Minor	1.589	A Major	1.364
half bar	A minor	0.17				0.34		0.10	0.48	0.41	1.00			A Minor	1.51	A Major	1.337
quarter	A minor	0.31	0.08	0.07		0.47	0.10	0.21	0.30	0.42	1.00	0.05		A Minor	1.78	A Major	1.555
eigth	A minor	0.30				0.46		0.38	0.96	0.36	1.00	0.05		A Minor	1.76	A Major	1.715

FIGURE 42 - TIME SCALE EFFECTS ON TRANSCRIPTION

All time scales resulted in the correct transcription but did affect the noise content. The two ways to analyse this is observing how much additional content is included in the chromagram as the time frame is decreased which shows how the short-time noise elements is filtered out with a longer time scale. This is a major concern for transcription accuracy as you do not want aspects such as drums to add any considerable content to the chromagram, which is the case in the full bar sample where only two additional notes are included compared to seven additional notes in the quarter sample.

The second method of observing the effect of time scaling is the correlation between the transcribed chord and the correlation between the second in line (the second highest probability) transcribed chord. As you decrease the time scale, the difference between these two decreases, which means the probability of incorrectly transcribing the chord increases. This is most evident for the time scale of an eighth bar, where the difference between the correlation of A minor and A major is only 0.045, an extremely small distance. This shows how averaging the analysis over the full length of the chord can increase the accuracy of transcription. This can be achieved through a reliable segmentation algorithm within the full automatic transcription process.

6.7 FULL PRODUCTION SONG WITHOUT EFFECTS

For this experiment, a completed song without any additional effects was recorded, manually segmented into full-bar time scales and then analysed using the same methods used previously. The no-effect song was first analysed to have a control analysis as jumping from isolated chords to a final song with effects could be too much of a leap to correctly observe the sources of error.

Given the song used for this experiment had four distinct sections, each systematically introducing more musical noise into the song, each section will be analysed separately. Starting with the first four bars where only percussion and electric guitar were used, the following data is extracted from the transcription process:

TABLE 19 - FIRST 4 CHORD TRANSCRIPTION ANALYSES

Chord	Transcription	Correlation
C Major	C Major	1.523
E Minor	E Minor	1.880
A Minor	A Minor	2.495
F Major	F Major	1.886

Here a 100% transcription was achieved, which suggests that percussion based noise can be minimised by using a large time scale as previously suggested. It is also noted that the range of correlation values are between, approximately, 1.5 and 2.5. This will be compared to future correlation values as noise is introduced into the sound file. Even though there are only four samples used here, useful information can still be extracted when comparing them due to the played chords being kept constant over them.

The next section of the song is analysed. This section includes the exact same chord sequence but with a bass guitar playing along with the root note of each chord.

TABLE 20 - SECOND SET OF 4 CHORD TRANSCRIPTION ANALYSES

Chord	Transcription	Correlation	Correlation with correct chord
C Major	C Major	1.345	
E Minor	E Major	1.936	1.928
A Minor	A Minor	2.684	
F Major	F Major	2.105	

Here an incorrect transcription was found, where the difference in the correlations has a difference of 0.01. This shows the danger of having additional musical content within the chromagram, where in this case the power value for note G=0.191 and G#=0.199 causes the incorrect transcription. In this case, musical theory could correct this error easily given the previous four chords have set the key.

The next four bars of the song introduce the vocal element which brings the implication that the vocals do not follow the root notes as often as the bass does. This means that a more dominant power spectrum will be added to notes which could change the transcription. Although the vocals will follow the key of the song, it does not have to be included in the three notes that construct a chord, and therefore is still considered noise to the chord transcription process. The data obtained from this is summarised below.

TABLE 21 - THIRD SET OF 4 CHORD TRANSCRIPTION ANALYSES

Chord	Transcription	Correlation	Correlation with correct chord
C Major	C Major	1.494	
E Minor	E Major	1.298	1.238
A Minor	A Minor	1.589	
F Major	F Major	1.503	

Here the same error was observed, with the difference between the correct and incorrect transcription being very minor. The power values were also lowered due to the additional power content from the vocal track which again suggests that the more noise added into the signal, the lower the correlation becomes.

The final section, again, had all the musical elements from the last section but with a different chord progression to ensure that the above samples give a realistic reflection of the noise. The following table was found when analysing this section.

TABLE 22 - FOURTH SET OF 4 CHORD TRANSCRIPTION ANALYSIS

Chord	Transcription	Correlation	Correlation with correct chord
C Major	C Augmented	1.836	1.742
G Major	G Major	1.514	
F Major	F Major	1.081	
G Major	G Major	1.085	

This sample displays the same behaviour as previously, suggesting that it was an accurate, yet limited, example of a full song. Above an incorrect transcription was observed with the same problem as previously mentioned. Namely other effects of the song added a component to a note that caused a slightly higher correlation with the equivalent chord within a different chord class. Music theory could be used here to correct this again.

From all the samples, it is observed that the rough correlation is approximately half the maximum that an exact matching chord should be to theory which is the problem that should be focused on through various methods in improving the current process of automatic transcription. Various methods which could possible decrease the effects of this noise will be discussed in the final chapter.

6.8 FULL PRODUCTION SONG WITH EFFECTS

To observe the implications of using musical effects to alter a songs audio qualities has on the final transcription, the same song as above was then adjusted to be more accurate to the industry standard of a final musical song. These samples will give the most accurate representation of the final audio product that would typically be used in the automatic transcription process.

Running the results for the transcription gives the following table. Here all 16 chords are included in the same table, with the corresponding transcription from the previous, 'no effects' samples for contrast on the right.

TABLE 23 - FULL EFFECTS TRANSCRIPTION RESULTS

Chord	Transcription	Correlation	Transcription (No effect)	Correlation
C Major	C Major	1.532	C Major	1.523
E Minor	E Minor	1.895	E Minor	1.880
A Minor	A Minor	2.508	A Minor	2.495
F Major	F Major	1.939	F Major	1.886
C Major	C Major	1.458	C Major	1.345
E Minor	E Major	1.715	E Major	1.936
A Minor	A Minor	1.904	A Minor	2.684
F Major	F Major	2.188	F Major	2.105
C Major	C Major	1.405	C Major	1.494
E Minor	E Minor	1.251	E Major	1.298
A Minor	A Minor	1.635	A Minor	1.589
F Major	F Major	1.464	F Major	1.503
C Major	C Major	1.888	C Augmented	1.836
G Major	G Major	1.702	G Major	1.514
F Major	F Minor	1.146	F Major	1.081
G Major	C Major	1.165	C Major	1.085

This data set yields interesting results which show that ideally adding effects into songs will improve the overall accuracy of the transcription. The red in the data table marks the incorrect transcriptions. However, in the effects laced song, only three incorrect transcriptions are found, one less than the 'no effects' song samples. The green above shows where the correlation with the chord transcribed increased from the 'no effects' data set. It is a desirable trait to improve the correlation as previously mentioned, where this increase can be attributed through the use of compression which effectively improves the power values of non-dominant notes which are inclusive in the chord and therefore increase the relationship between the data set and theory.

Again music theory could be used to correct two-thirds of the incorrect chords above (the E major and F minor, as they are not chords that are found within the scale used above), where the C major chord which cannot be fixed with theory due to its inclusion in the scale. There are possible methods for this error correction however and these will be discussed in chapter

7. This data set concludes the experimental data sets and shows the closest data set to the real world application of this song.

Since the design of the experiments throughout the project were designed to isolate and identify the noise sources that contribute to the incorrect transcription in an automatic process and, therefore, a brief overview of the noise content within a final piece of music is described in the next section.

6.9 Sources of Noise

A summary of the errors observed throughout this project are found in the following. The systematic introduction of noise into the audio files being transcribed gives the opportunity to correctly isolate and identify the sources of these noise elements. The first table simply states what noise sources are contained within the audio samples. The second table below shows an overview of the noise content within each data set and a loose rating from 1-3 given based on their effect on the data.

TABLE 24 - NOISE ELEMTNS INCLUDED IN EACH DATA SET

	Computation Time Vs. Accuracy	Instrumentation	A2D converter	Human element	Tone Alteration from recording process	Bass Power Content	Percussion	Non-Chord related musical notes	Musical Effects
Sine Wave Generator	X								
Synthetic Piano	X	X							
Electric Guitar	X	X	X	X					
Distorted Electric Guitar	X	X	X	X					
Acoustic Guitar	X	X	X	X	X				
Full Song (Without Effect)	X	X	X	X	X	X	X	X	
Full Song (With Effect)	X	X	X	X	X	X	X	X	X

TABLE 25 - EFFECTS OF NOISE CONTENT ON TRANCRIPTION ACCURACY

	Computation Time Vs. Accuracy	Instrumentation	A2D converter	Human error	Tone Alteration from recording	Bass Power Content	Percussion	Non-Chord related musical notes	Musical Effects
Sine Wave Generator	1								
Synthetic Piano	1	1							
Electric Guitar	1	2	1	3					
Distorted Electric Guitar	1	2	1	1					
Acoustic Guitar	1	3	1	3	2	3			
Full Song (Without Effect)	1	2	1	2	1	3	2	2	
Full Song (With Effect)	1	2	1	2	1	3	1	2	1

In the next few sections, a discussion of the implications of the different noise sources, their effect on the audio and possible means for decreasing their presence in the samples is found.

6.9.1 COMPUTATION TIME

As found in section 7.1, minor inaccuracies were introduced by the fast Fourier transform and chromagram function. The desire to translate the process of automatic transcription into a real time system means that the computational time is an extremely important factor to consider. Given the implications of having to process the audio, run the chromagram function, use the MLA and then display the transcription in a way that is readable and useable, reducing time it takes for each function can one day see the implementation into a portable consumer product. Here the accuracy versus speed argument, commonly discussed in many computation processes, is the main factor and is therefore dependant largely on the hardware able to be accessed. The best accuracy achieved by Dr M. Mauch utilised a super computer for its computations. This still required a time roughly equal to the song it was analysing (which is the maximum length of time for a real time system to be viable).

Means for reducing the effect of this noise source is to simply increase the number of sampling points the Fourier transform uses as previously discussed. Also increasing the hardware specifications of the machine you are using to analyse the song can contribute to a higher efficiency and allow for the optimum variable parameters to be used. This however is subject to future advancements in computing technology and not an issue that can be overcome in the near future, especially considering that any other methods for increasing accuracy would most likely result in increasing the computation requirements.

6.9.2 HUMAN ERROR

This source of noise should not be labelled as human error in the literal sense. There are several facets to this source and are related largely to the difference between the human brain and a computational device. It was clearly seen that when a human plays an instrument, the first occurrence of inaccuracy was observed in the isolated chords (Acoustic Guitar).This relates to the phenomenon where a listener can identify the same chord even though it is being played in different methods.

This however is not the case with a computer, as it can only analyse the data observed, and if a chord is played differently (which there are too many variations across all instruments to practically learn them to allow a computational device to recognise it as the same chord) the computer will take it as a different result. This large variability in the data means that there is no real method to overcome this issue unless a standard method of playing chords is enforced in recorded music. this would change the definition of music from being art to merely a means to become popular and commercialise off this fact (a trend not completely void in modern music). Its effect can be reduced however by assumptions made in other aspects of the complete process which means this source of noise will not become a dominant reason for incorrect transcriptions. Therefore focus should be given to the other sources found in this project.

6.9.3 INSTRUMENTATION NOISE

Instrumentation noise relates specifically to acoustic instruments but more generally to any source of musical audio. Ideally the spectrum qualities will match between the same instrument types which are rarely the case. This means that two instruments of the same type can have different frequency responses. This along with the changing of its frequency content in the post production stage of audio recordings means that little assumptions can be made to what frequency content will be in a song even if the instruments are known.

The more accurate assumptions that can be made to specific types of music (mainly different genres) will allow for the increase in overall transcription accuracy when solutions are discovered to overcome these. This means that if we can make an assumption as to the power values of frequencies within an audio sample, particularly focusing on removing the dominating effects a high powered frequency can have on the chromagram, an improvement in accuracy might be observed.

If a threshold is set to say 60%, the maximum peak value within the frequency spectrum, and all the frequencies above this is brought down to the same power level, some of these acoustic effects could be reduced. This will especially help in the cases where the bass-frequencies dominate the signal which reduces the other contributions of chord-related notes within the chromagram. An example of this is found in the diagram below, which is a frequency transform of the full production song with effects.

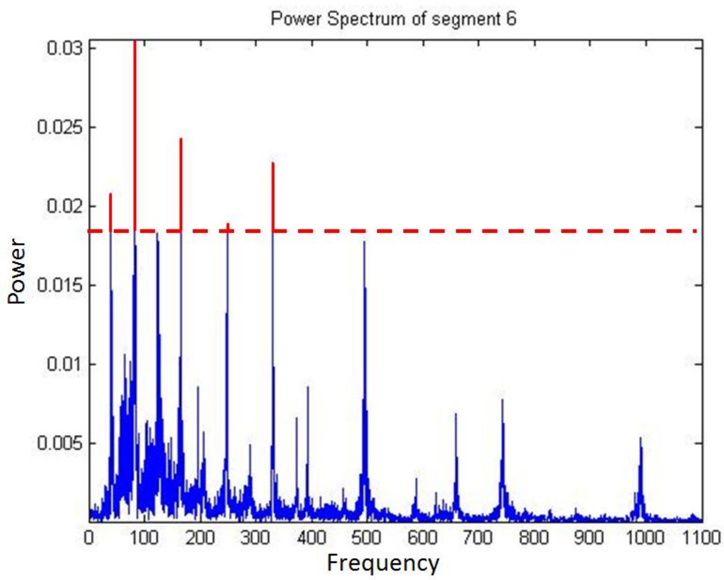

FIGURE 43 - THRESHOLD FREQUENCY ALTERATION EXAMPLE

Where with the threshold applied gives you the following graph.

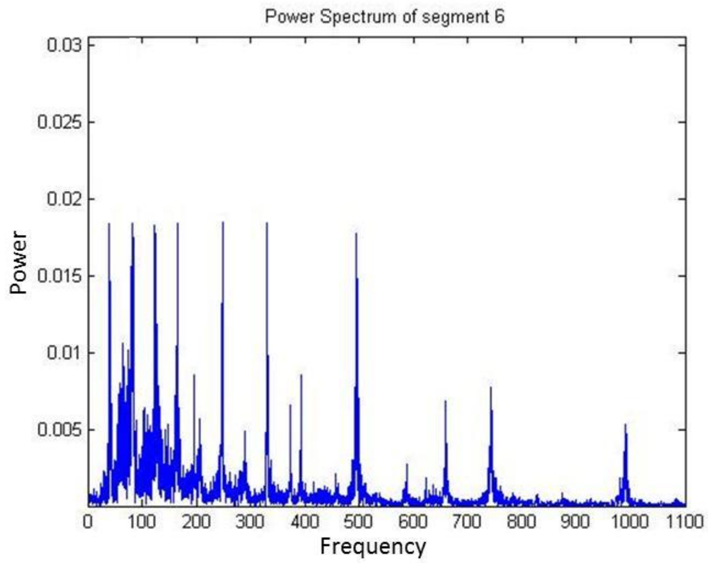

FIGURE 44 - RESULTING FREQUENCY SPECTRUM AFTER THRESHOLD IS APPLIED

Although further experimentation is needed to see if this would in fact increase the accuracy, or simply make non-chord related notes dominate the signal even more than previously.

If a standard and accurate method of reducing this source of noise can be found, a large increase in transcription accuracy should be observed due to the effect seen throughout the experiments where a non-chord note dominates the signal, all the other notes are scaled down due to the normalisation effect of the chromagram and errors are introduced. This should be the focus for future work.

6.9.4 Bass Frequencies

This is possibly the largest source of error within the whole process and slightly relates the acoustic errors above but should be considered differently due to its importance. As bass frequencies will naturally dominate the signal due to the computational time error discovered previously, having a high powered bass signal in a music piece can greatly decrease the correlation between the data and the theoretical chords.

This domination effect can be overcome by applying a method proposed by Dr Mauch (and utilised in his system), where you separate the audio data into two frequency ranges. This automated process will split the bass content from the treble content, and the separately analyse them. This also compliments musical theory and practical data where the bass spectrum often plays a vital role in identifying not only the key, but the chords. This was the bases for the improvements to the process Dr Mauch investigated, and showed a large increase in accuracy with his system [2].

This knowledge can again be used if additional information can be extracted from the song which relates to its genre, where certain assumptions can be made as to what instruments are more likely to feature within a song. This means that the audio data can be split in terms of frequency content even further into multiple ranges which correspond to the more likely instrumental classes (where it is known that what frequency range an electric bass covers compared to that of a double bass, compared to that of a cello etc.). This method should be easily implemented into a full process, but can add additional computation time if multiple frequency ranges are analysed. This is again subject to future advancements in hardware.

6.9.5 Musical Noise

Musical noise is classified as audio content within a song that interferes with the underlying chords being currently played. This is in comparison to non-musical noise such as background noise which will be filtered out by the chromagram function. Vocals, guitar solos, or simply multiple instruments being played simultaneously can add enough spectral content to a segment of audio to cause the correlation between multiple chords to become equal or incorrect and therefore an accurate transcription cannot be made.

This can be overcome again by frequency segmentation and separate analysis, with the final data being used to more accurately predict the chords. This will increase the accuracy if the correct musical theory is taken into consideration as to which frequency range will carry the dominant chord data (often the mid frequency which guitars and pianos are often constricted to in popular music). Upper partials also come in to play, where segmentation can remove a lot of this effect and if one frequency range carries a different chord profile than a lower one, assumptions can be made about the upper partials and a more likely chance of extracting the right chords can be achieved. This however is subject to experimental findings on this topic, where a whole project should be dedicated to this to correctly find the optimal method of segmentation to achieve an improved transcription rate.

6.10 Conclusion

Through careful analysis of the data found through the experimental procedures, the source and effect of different noise components that can typically be found in music was observed and analysed. Although the identification of all possible noise sources was not achieved, the major components which are typically found in all musical recordings (to some degree) were found. It should be noted that any automatic process will never be perfect since the variability of data that can be found in music is far too substantial to accurately work around all of them.

This report focuses on finding and analysing those noise sources which would be present in the largest possible pool of songs, and from these assumptions can be made relating to an automatic transcription process. But it is obvious that simply identifying the noise sources and their effect is only one aspect required to make use of the details of this report thus far. Suggestions of how the different noise sources can be reduced can also be given, but again this is only part of the final implementation of the results. Recommendations to the current automatic transcription process is therefore given in the next chapter, which takes all the results obtained in this report and applies it to realistically implementation in a final automatic transcription process which would see an increase in output accuracy.

7 Recommendations To The Current Automatic Transcription Process/Future work

Understanding where the noise sources within music originate from can allow us to modify the current methods, or add additional steps to the overall process that will help improve the overall accuracy of automatic music transcription. Since the most dominant noise elements have been identified in the previous chapter, this chapter will outline suggestions to the current process that would reduce those noise sources which can be controlled. In general, noise within the audio cannot be removed or altered, but rather steps taken in the process to reduce their effect or bypass them altogether. Several modifications or additions the processes are outlined below.

7.1 Extracting additional information

As previously mentioned, additional data can be extracted from audio files which, if used correctly, can increase the accuracy of an automatic process. This information is found within what is called the 'ID3 tag' which is associated to the '.mp3' file formats (the dominant file format of music today). Information that can be found in a typical tag (with example) is summarised in the table below:

TABLE 26 - INFORMATION INCLUDED IN ID3 TAG

Tag	Example
Title	The Thesis song
Artist	Anton Gouws
Album	Experimental work
Track Number	7
Year	2012
Genre	pop

This information is included in each song and is mainly used by software applications to sort and identify the tracks. This information is hard coded into the track at the time of final audio mix down and therefore cannot be edited after release. This is important when it comes to the initial assumptions made by the automatic transcription software, as this information can be used to adjust the overall process.

An example of this is using the genre to deduce the expected chords within the song. As previously stated genres such as 'rock' and 'metal' generally use power chords as their dominant chord structures and will most likely include distortion of some kind. Genres such as 'pop' or 'electronic' will typically use more synthetically generated sounds and also a more predicable chord progression can be expected. Assumptions to the tonal qualities can also be deduced from the genre and also year of recording (since older recordings tend to see an increase in background noise). Finding the optimal parameters for the variables within the automatic transcription process based on the genre would take trial and error (but should be based on the typical implications of the instruments usually used within that genre). If this information is not available, either user interaction is required to give a rough estimate of the genre, or simply left blank where default values are used. If this information is used correctly, and the correct alterations to the program are made based on those assumptions an increase in accuracy should be observed in the final product. This however will require a large amount of research put into what variables are optimised for a genre in general, and the main focus for doing this should be an automated learning process which identifies the important qualities within a large pool of songs and predicts the optimum variables that way. However, this is up to future work on this topic.

This however would be a good source of information about the song, as typical methods do not have any pre-made assumptions about songs which are a major factor to why inaccuracies are so hard to overcome. It would be uncommon to find a musical file that would typically be analysed using this process that does not include this information tag due to the extreme popularity of online media, where applications such as 'iTunes' and 'Spotify' would not allow a song to be included in their database without this information as it is used for automatic sorting of the media. Therefore it would be an extremely valuable aspect which should be utilised if 100% accuracy is to be achieved in the future.

7.2 PRE-DEFINED CHORD DATABASE

Any process of automatic transcription will use a pre-defined chord database in some fashion to match the data seen with the theoretical chord values. This not only allows easy use of theory but also sets the number of different chords which the software can analyse. Usually this database is constructed using theoretical values and is recorded in binary (1 for note inclusion, and 0 for note exclusion). However as found in the experimental work, it is rare that a chord played using non computer generated sounds will not exhibit a perfect correlation to theory.

This means that modifying the pre-defined chord data base such that it does not include purely binary values might increase the accuracy where a 'learning' approach can be used to obtain better base chord data entries to which the sampled data is compared to. This again can be incorporated in to the previous suggestion of extracting genre information and a pre-defined chord database for each genre can be used instead. This can be taken even further by applying a global learning approach where the chord database is learned as more and more music is analysed using the software. If enough songs are correctly analysed, the data is then used to adjust the current chord data base with a maximum likelihood algorithm, over time the accuracy of transcription should increase. This should however be confined to each genre as a global chord model will not yield the optimum results.

The only implication of altering the database to not include purely binary values is that simple matrix multiplication (as used in this thesis) cannot be used and a more complex MLA needs to be used which would likely see an increase in computational time needed which brings it back to the 'computational time vs. accuracy' argument.

7.3 FILTERING

Two approaches are suggested here relating to the word 'filtering', where either filtering out unwanted elements can be used or filtering the sample into multiple frequency ranges which can then be analysed separately.

The first suggestion is to reduce (or remove) certain frequency ranges which might interfere with the desired chord related information. Removing the higher spectrum which is less likely to contain musical instrumentation but rather upper partials from the musical instrumentation in the lower spectral range should see some improvement in the correlation between the data and the theory. Alternatively, removing some of the bass content could reduce the effect witnessed in the experiments where the bass frequencies dominate the chromagram. This reduces contributions from other musical notes and could also be done to not completely remove the bass qualities but simply reduce them in a way to negate the effects of the Fourier transform which has been shown to reduce the correlation with theory.

The second method suggested previously is segmenting the frequency content and then analysing them separately. The information obtained from both can then be compared and then used to show a more accurate representation of the musical chords. This method can be achieved by simply analysing the music with a band-pass filter ranging the frequencies that you want to analyse separately and then re-analysing the audio by shifting the cut-off frequencies higher values. Although as previously stated this will essentially multiply the amount of computational time needed to complete the transcription by the number of frequency ranges you are individually analysing. This means this is not a justifiable real time approach as this time and should only be seriously developed (in an individual user-based software package at least) when technology has allowed for computational time to reduce by several orders.

7.4 Applying Music Theory

The last section essentially overlays all the above suggestions, and should be a major focus for future work. The complex nature of music is the reason why reliable and accurate transcriptions have not been achieved yet as it is a difficult task to correctly input all the available musical theory into a software application such that it can be used efficiently. But regardless this would be an important aspect to focus on in terms of error detection and correction.

Predicting the chords of a song given the key would be a much simpler task as it would not improve the quality of the data obtained but rather reduce the number of possible chords to which it is being compared to. It was found that when a full song was analysed with the MLA, the correlation between an incorrect chord and the correct one was very small, to the order of 0.05 (which is a percentage difference of 1.6%). But if the key was known, the incorrect transcription would not be included in the data set to which the sample was compared to, and a correct transcription would have been achieved.

This however means that the key needs to be determined and can only be done by learning it as more and more chords are transcribed. Information such as genre (and also artists as the process is developed further) can also be used to limit the possible keys of the song. The song being analysed could have its key determined within the first few chord changes in a song, and then the song can be re-analysed or just corrections made automatically. This would see a large increase in accuracy if done correctly, and would allow an automatic process to be used on the more complex musical genres such as jazz which uses considerably more complicated chords, chord progressions, time signatures, and technicality when compared to more recent 'popular' music. Even if no other method for noise reduction or improvements to the current process is applied, using music theory for error correction would improve the overall accuracy of a working system by the largest degree.

8 CONCLUSION

Through the systematic introduction of noise which could potentially negatively affect the process of automatic transcription of music, the identification of multiple noise sources and their relative effect was achieved. The common dilemma of 'Time vs. Accuracy' was found to be an important contribution to noise, where advancements in computational power will be the only real solution. The human elements which can be found in most songs also contribute to incorrect transcriptions of chords, where there is no real solution to reducing this source of error.

Other errors identified include acoustic based frequency altering qualities introduced when using different instruments or microphones when recording the music, but these do not have a large effect on the data and therefore would take relatively little effort to reduce, or more commonly just ignored. Dominating bass frequencies show a large source of error which reduces the influence of other musical notes in the chromagram. This can be corrected by either filtering it out completely, or more appropriately analysing the song in more than one part, where the frequency of each part is divided into bass and treble (or more segments reducing the frequency range of a particular segment) which would not only reduce this effect, but allow more accurate transcriptions when applying musical theory.

However, the most important factor found throughout the experiments was the need to apply musical theory to the data to detect and correct errors in the transcription. Automatically determining the key in which the song is played in, and then using this information to correct transcriptions would potentially yield the best results in correct transcription rate, and therefore should be the focus of future work on this topic.

9 BIBLIOGRAPHY

[1] MathWorks, *MATLAB,* http://www.mathworks.com.au/products/matlab/?s_cid=wiki_matlab_2, 2011.

[2] M. Mauch, "Automatic Chord Transcription from Audio," University of London, London, 2010.

[3] A. Klapuri, Introduction to Music Transcription, Springer Science, 2006.

[4] H. J. Kallman, "Octave equivalence as measured by similarity ratings," *Perception & Psychophysics,* vol. 32, no. 1, pp. 37-49, 1982.

[5] V. T. M. Dictionary, "http://www.music.vt.edu/musicdictionary/," [Online]. [Accessed March 2012].

[6] D. Deutsch, "Music Recognition," 1969, pp. 300-307.

[7] M. S. S. A. A. a. E. G. Christopher Harte, "Symbolic representation of musical chords: A proposed text annotation," in *6th International Conference on Music Information Retrieval*, London, 2005.

[8] T. Fujishima, "Real time chord recognition of musical sound," *International computer music conference,* pp. 464-467, 1999.

[9] J. W. Ulrich, "The analysis and synthesis of jazz by computer," in *International Joint Conference on Artifcial Intelligence*, Los Altos, 1977.

[10] B. Pardo, "Algorithms for chordal analysis," *Computer Music Journal,* p. 27–49, 2002.

[11] X. Z. a. D. Gerhard., "Chord Recognition using Instrument Voicing Constraints.," 2008.

[12] F. J. Harris, "the use of windows for harmonic analysis with the discrete Fourier transform," in *IEEE*, 1978.

[13] E. Prestini, The Evolution of Applied Harmonic Analysis: Models of the Real World, 2004.

[14] E. Gomez, "Tonal Description of Audio Music Signals," PhD thesis, Universitat Pompeu Fabra, Barcelona, 2006.

[15] J. C. Brown, "Calculation of a constant Q spectral transform," *Journal of the Acoustical,* p. 89, 1991.

[16] J. P. B. a. J. Pickens, "A Robust Mid-level Representation for Harmonic Content," *6th International Conference on Music Information,* p. 304–311, 2005.

[17] C. H. a. M. Sandler, "Automatic chord identifcation using a quantised chromagram," *Audio Engineering Society,* 2005.

[18] J. W. a. J. L. Durrieu, "An HMM-based audio chord detection system attenuating the main," 2008.

[19] K. Noland, "Using Musical Structure to Enhance Automatic," in *Proceedings of the 10th*

International Conference on Music Information, Japan, 2009.

[20] M. toolbox, *MirChromagram,* MATLAB, 2011.

[21] J. C. a. J. Tukey, “An algorithm for the machine calculation of complex Fourier,” in *Mathematics of computation*, 1965.

[22] M. K. a. M. Omologo, “Phase-change based tuning for automatic chord,” 2009.

[23] E. G. a. P. Herrera, “The song remains the same: identifying versions of the same piece,” Canada, 2006.

[24] M. P. Ryynänen, “Automatic Transcription of Pitch Content in Music and Selected Applications,” Tampere University of Technology, 2008.

[25] J. P. a. J.-P. M. Matthias Varewyck, “A novel chroma representationof polyphonic music based on multiple pitch tracking techniques,” 2008.

[26] J. P. B. a. J. Pickens, “A Robust Mid-level Representation for Harmonic Content,” 2005.

[27] A. S. a. Y. Wang, “Key, chord, and rhythm tracking of popular music recordings.,” *Computer Music Journal,* pp. 75-86, 2005.

[28] J. L. D. a. G. R. JanWeil, “Automatic generation of lead sheets from polyphonic music signals,” 2009.

[29] J. P. a. T. Crawford, “Harmonic models for polyphonic music retrieval,” 2002.

[30] E. V. a. F. B. Ricardo Scholz, “Robust modeling of musical chord sequences using probabilistic n-grams,” 2009.

[31] H. P. a. G. Peeters, “Large-scale Study of Chord Estimation Algorithms Based on Chroma Representation and HMM,” 2007.

[32] A. S. a. D. Ellis, “Chord segmentation and recognition using EM-trained hidden,” in *4th International Conference on Music Information Retrieval*, Baltimore, USA, 2003.

[33] L. P. C. K. a. I. F. John Ashley Burgoyne, “A cross-validated study of modelling strategies for automatic chord recognition in audio,” Austria, 2007.

[34] B. S. a. S. Jeng, “Multi-timbre chord classification using wavelet transform and self-organized,” 2001.

[35] F. P. a. J.-P. B. Giordano Cabral, “Automatic X traditional descriptor extraction: the case of chord recognition,” 2005.

[36] reFX, *Nexus Studio Synthesizer Plugin,* ReFx, 2010.

[37] Steinberg, *Cubase 5.1 DAW,* Steinberg, 2010.

[38] T. S. Anderson, “ELECTRICAL PICKUP FOR A STRINGED MUSICAL INSTRUMENT”. Patent 5168117, 14 January 1991.

[39] M-Audio, *Audiphile 2496 sound card*.

[40] Stenberg, *Groove-Agent ONE,* Cubase 5.1, 2010.

[41] D. Carlton, "I analyzed the chords of 1300 popular songs for patterns. This is what I found," Hook Theory, 6 June 2012. [Online]. Available: http://www.hooktheory.com/blog/i-analyzed-the-chords-of-1300-popular-songs-for-patterns-this-is-what-i-found/. [Accessed 2nd October 2012].

[42] J. C. a. J. Tukey, "An algorithm for the machine calculation of complex Fourier," in *Mathematics of computation*, 1965, pp. 297-301.

[43] H. P. a. G. Peeters, "Large-scale Study of Chord Estimation Algorithms," in *International Workshop on Content-Based*, 2007, pp. 53-60.

[44] M. K. a. M. Omologo, "Phase-change based tuning for automatic chord," in *12th International Conference on Digital Audio Effects*, Como, Italy, 2009.

[45] R. S. a. G. Ramalho, "Recognizing complex chords from midi," in *9th International Conference on Music Information*, Philadelphia, USA, 2008.

10 Appendix

10.1 Theoretical chord database

MAJOR	C	C#	D	D#	E	F	F#	G	G#	A	A#	B
C	1				1			1				
C#		1				1			1			
D			1				1			1		
D#				1				1			1	
E					1				1			1
F	1					1				1		
F#		1					1				1	
G			1					1				1
G#	1			1					1			
A		1			1					1		
A#			1			1					1	
B				1			1					1

FIGURE 45 - MAJOR CHORDS THEORY

MINOR	C	C#	D	D#	E	F	F#	G	G#	A	A#	B
C	1			1				1				
C#		1			1				1			
D			1			1				1		
D#				1			1				1	
E					1			1				1
F	1					1			1			
F#		1					1			1		
G			1					1			1	
G#				1					1			1
A	1				1					1		
A#		1				1					1	
B			1				1					1

FIGURE 46 - MINOR CHORDS THEORY

AUGMENTED	C	C#	D	D#	E	F	F#	G	G#	A	A#	B
C	1				1				1			
C#		1				1				1		
D			1				1				1	
D#				1				1				1
E	1				1				1			
F		1				1				1		
F#			1				1				1	
G				1				1				1
G#	1				1				1			
A		1				1				1		
A#			1				1				1	
B				1				1				1

FIGURE 47 - AUGMENTED TRIAD CHORDS THEORY

DIMINISHED	C	C#	D	D#	E	F	F#	G	G#	A	A#	B
C	1			1			1					
C#		1			1			1				
D			1			1			1			
D#				1			1			1		
E					1			1			1	
F						1			1			1
F#	1						1			1		
G		1						1			1	
G#			1						1			1
A	1			1						1		
A#		1			1						1	
B			1			1						1

FIGURE 48 - DIMINISHED TRIAD CHORDS THEORY

10.2 Sine Waves

MAJOR	C	C#	D	D#	E	F	F#	G	G#	A	A#	B
C	1.00				0.81			0.67				
C#		1.00				0.76			0.67			
D			1.00				0.79			0.66		
D#				1.00				0.78			0.65	
E					1.00				0.81			0.66
F	0.72					1.00				0.82		
F#		0.67					1.00				0.77	
G			0.67					1.00				0.79
G#	0.81			0.64					1.00			
A		0.79			0.66					1.00		
A#			0.80			0.67					1.00	
B				0.79			0.67					1.00

FIGURE 49 - MAJOR CHORDS WITH SYNTHETIC SINE WAVES

MINOR	C	C#	D	D#	E	F	F#	G	G#	A	A#	B
C	1.00			0.85				0.68				
C#		1.00			0.83				0.68			
D			1.00			0.85				0.67		
D#				1.00			0.84				0.70	
E					1.00			0.88				0.68
F	0.67					1.00			0.84			
F#		0.67					1.00			0.84		
G			0.65					1.00			0.85	
G#				0.66					1.00			0.82
A	0.87				0.69					1.00		
A#		0.81				0.65					1.00	
B			0.84				0.66					1.00

FIGURE 50 - MINOR CHORDS WITH SYNTHETIC SINE WAVES

AUGMENTED	C	C#	D	D#	E	F	F#	G	G#	A	A#	B
C	1.00				0.81				0.65			
C#		1.00				0.78				0.63		
D			1.00				0.79				0.63	
D#				1.00				0.79				0.61
E	0.66				1.00				0.81			
F		0.63				1.00				0.80		
F#			0.62				1.00				0.80	
G				0.62				1.00				0.78
G#	0.81				0.62				1.00			
A		0.79				0.62				1.00		
A#			0.77				0.61				1.00	
B				0.80				0.63				1.00

FIGURE 51 - AUGMENTED TRIAD CHORDS WITH SYNTHETIC SINE WAVES

DIMINISHED	C	C#	D	D#	E	F	F#	G	G#	A	A#	B
C	1.00			0.86			0.72					
C#		1.00			0.84			0.73				
D			1.00			0.86			0.72			
D#				1.00			0.83			0.70		
E					1.00			0.86			0.74	
F						1.00			0.84			0.70
F#	0.73						1.00			0.84		
G		0.69						1.00			0.85	
G#			0.70						1.00			0.83
A	0.87			0.71						1.00		
A#		0.81			0.69						1.00	
B			0.84			0.71						1.00

FIGURE 52 - DIMINISHED TRIAD CHORDS WITH SYNHETIC SINE WAVES

10.3 Piano Synthesizer

MAJOR	C	C#	D	D#	E	F	F#	G	G#	A	A#	B
C	0.72				1.00			0.65				0.02
C#		0.87				1.00			0.88			
D			1.00				0.65			0.55		
D#				1.00				0.62			0.47	
E					1.00				0.54			0.48
F	0.67					1.00				0.86		
F#		0.58					1.00				0.75	
G			0.42					1.00				0.80
G#	0.77			0.45					1.00			
A		0.68			0.39					1.00		
A#			0.52			0.23					1.00	
B				0.49			0.19					1.00

FIGURE 53 - PIANO MAJOR CHORDS

MINOR	C	C#	D	D#	E	F	F#	G	G#	A	A#	B
C	0.63			1.00				0.69			0.02	
C#		0.58			1.00				0.53			0.02
D			1.00			0.59				0.57		
D#				1.00			0.65				0.48	
E					1.00			0.61				0.48
F	0.60					1.00			0.78			
F#		0.59					1.00			0.88		
G			0.43					1.00			0.79	
G#				0.43					1.00			0.89
A	0.78				0.38					1.00		
A#		0.76				0.22					1.00	
B			0.55				0.19					1.00

FIGURE 54 - PAINO MINOR CHORDS

AUGMENTED	C	C#	D	D#	E	F	F#	G	G#	A	A#	B
C	0.72				1.00			0.04	0.55			
C#		0.84				1.00			0.05	0.79		
D			1.00				0.68			0.03	0.46	
D#				1.00				0.63			0.03	0.48
E	0.42				1.00				0.55			
F		0.52				1.00				0.75		
F#			0.42				1.00				0.76	
G				0.40				1.00				0.79
G#	0.78				0.37				1.00			
A		0.68				0.19				1.00		
A#			0.51				0.22				1.00	
B				0.50				0.13				1.00

FIGURE 55 - PIANO AUGMENTED TRIAD CHORDS

DIMINISHED	C	C#	D	D#	E	F	F#	G	G#	A	A#	B
C	0.63			1.00			0.56					
C#		0.53			1.00			0.60	0.03			0.02
D			1.00			0.62			0.55	0.03		
D#				1.00			0.64			0.54		
E					1.00			0.60			0.45	
F						1.00			0.80			0.73
F#	0.69						1.00			0.87		
G		0.61						1.00			0.80	
G#			0.48						1.00			0.88
A	0.78			0.43						1.00		
A#		0.76			0.42						1.00	
B			0.55			0.20						1.00

FIGURE 56 - PIANO DIMINISHED TRIAD CHORDS

10.4 Electric Guitar Clean

MAJOR	C	C#	D	D#	E	F	F#	G	G#	A	A#	B
C	1.00				0.70			0.21				
C#		1.00				0.41			0.07			
D			1.00				0.35			0.10		
D#				1.00				0.69			0.61	
E					0.72				0.07			1.00
F	1.00					0.66				0.17		
F#		1.00					0.46				0.14	
G			1.00					0.32				0.48
G#	0.75			0.92					1.00			
A		0.55			1.00					0.79		
A#			0.43			0.81					1.00	
B				0.45			1.00					0.93

FIGURE 57 - ELECTRIC GUITAR MAJOR CHORD

MINOR	C	C#	D	D#	E	F	F#	G	G#	A	A#	B
C	1.00			0.70				0.55				
C#		0.67			1.00				0.04			
D			1.00			0.46				0.32		
D#				1.00			0.68				0.42	
E					0.76							1.00
F	1.00					0.47			0.05			
F#		1.00					0.50			0.20		
G			1.00					0.42			0.19	
G#				0.98					1.00			0.30
A	0.45				0.83					1.00		
A#		0.73				0.92					1.00	
B			0.46				0.37					1.00

FIGURE 58 - ELECTRIC GUITAR CLEAN MINOR CHORDS

AUGMENTED	C	C#	D	D#	E	F	F#	G	G#	A	A#	B
C	0.91				1.00				0.57			
C#		1.00				0.07				0.19		
D			1.00				0.33				0.24	
D#				0.48				0.37				1.00
E	0.92				1.00				0.27			
F		0.44				1.00				0.29		
F#			1.00								0.81	
G				0.75				1.00				0.61
G#	1.00				1.00				0.12			
A		0.62				0.77				1.00		
A#			1.00								0.61	
B				0.31				0.15				1.00

FIGURE 59 - ELECTRIC GUITAR CLEAN AUGMENTED TRIAD CHORDS

DIMINISHED	C	C#	D	D#	E	F	F#	G	G#	A	A#	B
C	0.78			1.00			0.51					
C#		0.80			1.00			0.50				
D			1.00			0.51			0.35			
D#				1.00			0.85			0.33		
E					0.38			1.00			0.50	
F						0.53			0.36			1.00
F#	1.00						0.60			0.23		
G		0.43						1.00				
G#			1.00						0.33			0.86
A	0.53			0.63						1.00		
A#		1.00			0.65						0.63	
B			0.32			0.18						1.00

FIGURE 60 - ELECTRIC GUITAR CLEAN DIMINISHED TRIAD CHORDS

10.5 ACOUSTIC GUITAR

MAJOR	C	C#	D	D#	E	F	F#	G	G#	A	A#	B
C	1.00				0.48			0.19				
C#		1.00				0.56			0.08			
D			1.00				0.15			0.06		
D#				0.46				1.00			0.54	
E					0.56				0.09			1.00
F	1.00					0.33				0.16		
F#		1.00					0.31				0.22	
G			1.00					0.49				1.00
G#	1.00			0.76					0.77			
A		0.28			1.00					1.00		
A#			0.19			0.75					1.00	
B				0.11			0.46					1.00

FIGURE 61 - ACOUSTIC GUITAR MAJOR CHORDS

MINOR	C	C#	D	D#	E	F	F#	G	G#	A	A#	B
C	1.00			0.65				0.44				
C#		0.39			1.00				0.20			
D			1.00			0.06				0.14		
D#				0.26			1.00				0.50	
E					0.54			0.17				1.00
F	1.00					0.70			0.05			
F#		1.00					0.33			0.02		
G			1.00					0.11			0.02	
G#				1.00					1.00			0.22
A	0.17				0.62					1.00		
A#		0.21				0.51					1.00	
B			0.16				0.49					1.00

FIGURE 62 - ACOUSTIC GUITAR MINOR CHORDS

AUGMENTED	C	C#	D	D#	E	F	F#	G	G#	A	A#	B
C	1.00				0.38				0.04			
C#		1.00				0.36						
D			1.00				0.08				0.26	
D#				0.49				0.60				1.00
E	0.80				1.00							
F		0.42				1.00				0.11		
F#			1.00				0.12				0.92	
G				0.38				1.00				
G#	1.00				0.53							
A		0.29				0.53				1.00		
A#			0.93								1.00	
B				0.56				0.30				1.00

FIGURE 63 - ACOUSTIC GUITAR AUGMENTED TRIAD CHORDS

DIMINISHED	C	C#	D	D#	E	F	F#	G	G#	A	A#	B
C	0.21			1.00			0.17					
C#		0.31			1.00			0.11				
D						1.00			0.21			
D#				1.00			0.17					
E					1.00			0.11			0.66	
F						1.00			0.21			0.58
F#	0.21						0.17					
G		0.31						1.00			0.66	
G#			0.21						0.68			1.00
A	0.21			1.00								
A#		0.31			1.00						0.66	
B			0.23			1.00						0.13

FIGURE 64 - ACOUSTIC GUITAR DIMINISHED TRIAD CHORDS

10.6 ELECTRIC GUITAR DISTORTED

Theory	C	C#	D	D#	E	F	F#	G	G#	A	A#	B
Power chord E					1.00							1.00
Power chord G			1.00					1.00				
Power chord A					1.00					1.00		

FIGURE 65 - DISTORED ELECTRIC GUITAR THEORY

Data	C	C#	D	D#	E	F	F#	G	G#	A	A#	B
Power chord E					0.99							1.00
Power chord G			1.00					0.99				
Power chord A					1.00					0.97		

FIGURE 66 - DISTORTED ELECTRIC GUITAR

10.7 Full Production Song

Segment	Percussion	Electric Guitar	Bass	Vocal	Chord	C	C#	D	D#	E	F	F#	G	G#	A	A#	B
1	x	x			C Major	1.00				0.33			0.19	0.06			0.02
2	x	x			E Minor	0.08				0.79		0.06	0.09	0.06			1.00
3	x	x			A Minor	0.88	0.05	0.07	0.04	0.62			0.14	0.14	1.00		0.04
4	x	x			F Major	1.00	0.03				0.69		0.02	0.04	0.19		
5	x	x	x		C Major	1.00				0.25			0.10	0.08			
6	x	x	x		E Minor	0.14	0.08	0.09		0.74		0.07	0.19	0.20		0.12	1.00
7	x	x	x		A Minor	0.72	0.08			1.00			0.15	0.26	0.96		0.10
8	x	x	x		F Major	1.00					0.90		0.05	0.18	0.21		0.08
9	x	x	x	x	C Major	1.00	0.26			0.30			0.19	0.12	0.11	0.12	0.45
10	x	x	x	x	E Minor	0.10				0.24				0.06			1.00
11	x	x	x	x	A Minor	0.23				0.36			0.29	0.33	1.00		
12	x	x	x	x	F Major	1.00					0.31			0.11	0.19		0.16
13	x	x	x	x	C Major	0.62				1.00			0.12	0.21			0.16
14	x	x	x	x	G Major	0.06		1.00		0.99			0.12	0.11	0.15		0.39
15	x	x	x	x	F Major	1.00					0.08						
16	x	x	x	x	G Major	1.00	0.18	0.26					0.08			0.15	0.46

FIGURE 67 - FULL PRODUCTION WITH NO EFFECTS DATA SET

Segment	Percussion	Electric Guitar	Bass	Vocal	Chord	C	C#	D	D#	E	F	F#	G	G#	A	A#	B
1	x	x			C Major	1.00				0.35			0.18				0.01
2	x	x			E Minor	0.11				0.77		0.08	0.12	0.07			1.00
3	x	x			A Minor	0.80				0.63				0.08	1.00		
4	x	x			F Major	1.00					0.74		0.04		0.19		
5	x	x	x		C Major	1.00				0.22			0.23	0.05			
6	x	x	x		E Minor	0.30	0.09	0.05		0.54			0.14	0.17			1.00
7	x	x	x		A Minor	0.24				0.66				0.26	1.00		
8	x	x	x		F Major	0.99					1.00		0.18	0.09	0.19		
9	x	x	x	x	C Major	1.00				0.22			0.18				0.12
10	x	x	x	x	E Minor	0.06				0.25							1.00
11	x	x	x	x	A Minor	0.20				0.43			0.18		1.00		
12	x	x	x	x	F Major	1.00					0.30				0.16		
13	x	x	x	x	C Major	1.00				0.70			0.18	0.06			0.09
14	x	x	x	x	G Major			1.00		0.75			0.21		0.08		0.49
15	x	x	x	x	F Major	1.00					0.14						
16	x	x	x	x	G Major	1.00	0.09	0.45					0.16				0.37

FIGURE 68 - FULL PRODUCTION SONG WITH EFFECTS

10.8 MATLAB Code Used

Program that transcribes chords from the data presented in the tables above. Uses the MLA for transcription in MATLAB written by Anton Gouws. Sine wave data entered into this program, where the variables are simply changed to reflect the data of the required data set to be analysed

```
% Sine_waves shows relavent transcriptions of the collected data relating
% to generated sinusoidal musical chords.
% There is no input, as all data which is analysed is found within the
% program itself, in matrix form denoted by the variable Maj, Min, Aug, and
Dim.
%
% Written by Anton Gouws 2012


function [Maj_acc,Maj_ave,Min_acc,Min_ave,Aug_acc,Aug_ave]=Sine_Waves()

Maj =
[1,0,0,0,0.810000000000000,0,0,0.670000000000000,0,0,0,0;0,1,0,0,0,0.760000
000000000,0,0,0.670000000000000,0,0,0;0,0,1,0,0,0,0.790000000000000,0,0,0.6
60000000000000,0,0;0,0,0,1,0,0,0,0.780000000000000,0,0,0.650000000000000,0;
0,0,0,0,1,0,0,0,0.810000000000000,0,0,0.660000000000000;0.720000000000000,0
,0,0,0,1,0,0,0,0.820000000000000,0,0;0,0.670000000000000,0,0,0,0,1,0,0,0,0.
770000000000000,0;0,0,0.670000000000000,0,0,0,0,1,0,0,0,0.790000000000000;0
.810000000000000,0,0,0.640000000000000,0,0,0,0,1,0,0,0;0,0.790000000000000,
0,0,0.660000000000000,0,0,0,0,1,0,0;0,0,0.800000000000000,0,0,0.67000000000
0000,0,0,0,0,1,0;0,0,0,0.790000000000000,0,0,0.670000000000000,0,0,0,0,1;];
Min =
[1,0,0,0.850000000000000,0,0,0,0.680000000000000,0,0,0,0;0,1,0,0,0.83000000
0000000,0,0,0,0.680000000000000,0,0,0;0,0,1,0,0,0.850000000000000,0,0,0,0.6
70000000000000,0,0;0,0,0,1,0,0,0.840000000000000,0,0,0,0.700000000000000,0;
0,0,0,0,1,0,0,0.880000000000000,0,0,0,0.680000000000000;0.670000000000000,0
,0,0,0,1,0,0,0.840000000000000,0,0,0;0,0.670000000000000,0,0,0,0,1,0,0,0.84
0000000000000,0,0;0,0,0.650000000000000,0,0,0,0,1,0,0,0.850000000000000,0;0
,0,0,0.660000000000000,0,0,0,0,1,0,0,0.820000000000000;0.870000000000000,0,
0,0,0.690000000000000,0,0,0,0,1,0,0;0,0.810000000000000,0,0,0,0.65000000000
0000,0,0,0,0,1,0;0,0,0.840000000000000,0,0,0,0.660000000000000,0,0,0,0,1;];
Aug =
[1,0,0,0,0.810000000000000,0,0,0,0.650000000000000,0,0,0;0,1,0,0,0,0.780000
000000000,0,0,0,0.630000000000000,0,0;0,0,1,0,0,0,0.790000000000000,0,0,0,0
.630000000000000,0;0,0,0,1,0,0,0,0.790000000000000,0,0,0,0.610000000000000;
0.660000000000000,0,0,0,1,0,0,0,0.810000000000000,0,0,0;0,0.630000000000000
,0,0,0,1,0,0,0,0.800000000000000,0,0;0,0,0.620000000000000,0,0,0,1,0,0,0,0.
800000000000000,0;0,0,0,0.620000000000000,0,0,0,1,0,0,0,0.780000000000000;0
.810000000000000,0,0,0,0.620000000000000,0,0,0,1,0,0,0;0,0.790000000000000,
0,0,0,0.620000000000000,0,0,0,1,0,0;0,0,0.770000000000000,0,0,0,0.610000000
000000,0,0,0,1,0;0,0,0,0.800000000000000,0,0,0,0.630000000000000,0,0,0,1;];
Dim =
[1,0,0,0.860000000000000,0,0,0.720000000000000,0,0,0,0,0;0,1,0,0,0.84000000
0000000,0,0,0.730000000000000,0,0,0,0;0,0,1,0,0,0.860000000000000,0,0,0.720
000000000000,0,0,0;0,0,0,1,0,0,0.830000000000000,0,0,0.700000000000000,0,0;
0,0,0,0,1,0,0,0.860000000000000,0,0,0.740000000000000,0;0,0,0,0,0,1,0,0,0.8
40000000000000,0,0,0.700000000000000;0.730000000000000,0,0,0,0,0,1,0,0,0.84
0000000000000,0,0;0,0.690000000000000,0,0,0,0,0,1,0,0,0.850000000000000,0;0
,0,0.700000000000000,0,0,0,0,0,1,0,0,0.830000000000000;0.870000000000000,0,
0,0.710000000000000,0,0,0,0,0,1,0,0;0,0.810000000000000,0,0,0.690000000000000
00,0,0,0,0,0,1,0;0,0,0.840000000000000,0,0,0.710000000000000,0,0,0,0,0,1;];

CMaj=zeros(1,12);CsMaj=zeros(1,12);DMaj=zeros(1,12);DsMaj=zeros(1,12);EMaj=
zeros(1,12);FMaj=zeros(1,12);FsMaj=zeros(1,12);GMaj=zeros(1,12);GsMaj=zeros
```

```
(1,12);AMaj=zeros(1,12);AsMaj=zeros(1,12);BMaj=zeros(1,12);
CMin=zeros(1,12);CsMin=zeros(1,12);DMin=zeros(1,12);DsMin=zeros(1,12);EMin=
zeros(1,12);FMin=zeros(1,12);FsMin=zeros(1,12);GMin=zeros(1,12);GsMin=zeros
(1,12);AMin=zeros(1,12);AsMin=zeros(1,12);BMin=zeros(1,12);
CAug=zeros(1,12);CsAug=zeros(1,12);DAug=zeros(1,12);DsAug=zeros(1,12);EAug=
zeros(1,12);FAug=zeros(1,12);FsAug=zeros(1,12);GAug=zeros(1,12);GsAug=zeros
(1,12);AAug=zeros(1,12);AsAug=zeros(1,12);BAug=zeros(1,12);
CDim=zeros(1,12);CsDim=zeros(1,12);DDim=zeros(1,12);DsDim=zeros(1,12);EDim=
zeros(1,12);FDim=zeros(1,12);FsDim=zeros(1,12);GDim=zeros(1,12);GsDim=zeros
(1,12);ADim=zeros(1,12);AsDim=zeros(1,12);BDim=zeros(1,12);

Maj_accuracy_maj=0;
Maj_out_ave=0;
Min_accuracy_Min=0;
Min_out_ave=0;
Aug_accuracy_Aug=0;
Aug_out_ave=0;
Dim_accuracy_Dim=0;
Dim_out_ave=0;

M = 0;
N = 0;
O = 0;
P = 0;

for i = 1:12
    CAug(1,i)=Aug(1,i);
    CsAug(1,i)=Aug(2,i);
    DAug(1,i)=Aug(3,i);
    DsAug(1,i)=Aug(4,i);
    EAug(1,i)=Aug(5,i);
    FAug(1,i)=Aug(6,i);
    FsAug(1,i)=Aug(7,i);
    GAug(1,i)=Aug(8,i);
    GsAug(1,i)=Aug(9,i);
    AAug(1,i)=Aug(10,i);
    AsAug(1,i)=Aug(11,i);
    BAug(1,i)=Aug(12,i);
end
for i = 1:12
    CDim(1,i)=Dim(1,i);
    CsDim(1,i)=Dim(2,i);
    DDim(1,i)=Dim(3,i);
    DsDim(1,i)=Dim(4,i);
    EDim(1,i)=Dim(5,i);
    FDim(1,i)=Dim(6,i);
    FsDim(1,i)=Dim(7,i);
    GDim(1,i)=Dim(8,i);
    GsDim(1,i)=Dim(9,i);
    ADim(1,i)=Dim(10,i);
    AsDim(1,i)=Dim(11,i);
    BDim(1,i)=Dim(12,i);
end
for i = 1:12
    CMaj(1,i)=Maj(1,i);
    CsMaj(1,i)=Maj(2,i);
    DMaj(1,i)=Maj(3,i);
    DsMaj(1,i)=Maj(4,i);
    EMaj(1,i)=Maj(5,i);
    FMaj(1,i)=Maj(6,i);
    FsMaj(1,i)=Maj(7,i);
```

```
    GMaj(1,i)=Maj(8,i);
    GsMaj(1,i)=Maj(9,i);
    AMaj(1,i)=Maj(10,i);
    AsMaj(1,i)=Maj(11,i);
    BMaj(1,i)=Maj(12,i);
end
for i = 1:12
    CMin(1,i)=Min(1,i);
    CsMin(1,i)=Min(2,i);
    DMin(1,i)=Min(3,i);
    DsMin(1,i)=Min(4,i);
    EMin(1,i)=Min(5,i);
    FMin(1,i)=Min(6,i);
    FsMin(1,i)=Min(7,i);
    GMin(1,i)=Min(8,i);
    GsMin(1,i)=Min(9,i);
    AMin(1,i)=Min(10,i);
    AsMin(1,i)=Min(11,i);
    BMin(1,i)=Min(12,i);
end


[chord_class_string,out] = Full_Analysis_2(CMaj);
if strcmp(chord_class_string,'C Major')
    Maj_accuracy_maj=Maj_accuracy_maj + 1;
    Maj_out_ave=Maj_out_ave+out;
    M = M+1;
end
[chord_class_string,out] = Full_Analysis_2(CsMaj);
if strcmp(chord_class_string,'C# Major')
    Maj_accuracy_maj=Maj_accuracy_maj + 1;
    Maj_out_ave=Maj_out_ave+out;
    M = M+1;
end
[chord_class_string,out] = Full_Analysis_2(DMaj);
if strcmp(chord_class_string,'D Major')
    Maj_accuracy_maj=Maj_accuracy_maj + 1;
    Maj_out_ave=Maj_out_ave+out;
    M = M+1;
end
[chord_class_string,out] = Full_Analysis_2(DsMaj);
if strcmp(chord_class_string,'D# Major')
    Maj_accuracy_maj=Maj_accuracy_maj + 1;
    Maj_out_ave=Maj_out_ave+out;
    M = M+1;
end
[chord_class_string,out] = Full_Analysis_2(EMaj);
if strcmp(chord_class_string,'E Major')
    Maj_accuracy_maj=Maj_accuracy_maj + 1;
    Maj_out_ave=Maj_out_ave+out;
    M = M+1;
end
[chord_class_string,out] = Full_Analysis_2(FMaj);
if strcmp(chord_class_string,'F Major')
    Maj_accuracy_maj=Maj_accuracy_maj + 1;
    Maj_out_ave=Maj_out_ave+out;
    M = M+1;
end
[chord_class_string,out] = Full_Analysis_2(FsMaj);
if strcmp(chord_class_string,'F# Major')
    Maj_accuracy_maj=Maj_accuracy_maj + 1;
```

```
    Maj_out_ave=Maj_out_ave+out;
    M = M+1;
end
[chord_class_string,out] = Full_Analysis_2(GMaj);
if strcmp(chord_class_string,'G Major')
    Maj_accuracy_maj=Maj_accuracy_maj + 1;
    Maj_out_ave=Maj_out_ave+out;
    M = M+1;
end
[chord_class_string,out] = Full_Analysis_2(GsMaj);
if strcmp(chord_class_string,'G# Major')
    Maj_accuracy_maj=Maj_accuracy_maj + 1;
    Maj_out_ave=Maj_out_ave+out;
    M = M+1;
end
[chord_class_string,out] = Full_Analysis_2(AMaj);
if strcmp(chord_class_string,'A Major')
    Maj_accuracy_maj=Maj_accuracy_maj + 1;
    Maj_out_ave=Maj_out_ave+out;
    M = M+1;
end
[chord_class_string,out] = Full_Analysis_2(AsMaj);
if strcmp(chord_class_string,'A# Major')
    Maj_accuracy_maj=Maj_accuracy_maj + 1;
    Maj_out_ave=Maj_out_ave+out;
    M = M+1;
end
[chord_class_string,out] = Full_Analysis_2(BMaj);
if strcmp(chord_class_string,'B Major')
    Maj_accuracy_maj=Maj_accuracy_maj + 1;
    Maj_out_ave=Maj_out_ave+out;
    M = M+1;
end
[chord_class_string,out] = Full_Analysis_2(CMin);
if strcmp(chord_class_string,'C Minor')
    Min_accuracy_Min=Min_accuracy_Min + 1;
    Min_out_ave=Min_out_ave+out;
    N = N+1;
end
[chord_class_string,out] = Full_Analysis_2(CsMin);
if strcmp(chord_class_string,'C# Minor')
    Min_accuracy_Min=Min_accuracy_Min + 1;
    Min_out_ave=Min_out_ave+out;
    N = N+1;
end
[chord_class_string,out] = Full_Analysis_2(DMin);
if strcmp(chord_class_string,'D Minor')
    Min_accuracy_Min=Min_accuracy_Min + 1;
    Min_out_ave=Min_out_ave+out;
    N = N+1;
end
[chord_class_string,out] = Full_Analysis_2(DsMin);
if strcmp(chord_class_string,'D# Minor')
    Min_accuracy_Min=Min_accuracy_Min + 1;
    Min_out_ave=Min_out_ave+out;
    N = N+1;
end
[chord_class_string,out] = Full_Analysis_2(EMin);
if strcmp(chord_class_string,'E Minor')
    Min_accuracy_Min=Min_accuracy_Min + 1;
    Min_out_ave=Min_out_ave+out;
```

```
    N = N+1;
end
[chord_class_string,out] = Full_Analysis_2(FMin);
if strcmp(chord_class_string,'F Minor')
    Min_accuracy_Min=Min_accuracy_Min + 1;
    Min_out_ave=Min_out_ave+out;
    N = N+1;
end
[chord_class_string,out] = Full_Analysis_2(FsMin);
if strcmp(chord_class_string,'F# Minor')
    Min_accuracy_Min=Min_accuracy_Min + 1;
    Min_out_ave=Min_out_ave+out;
    N = N+1;
end
[chord_class_string,out] = Full_Analysis_2(GMin);
if strcmp(chord_class_string,'G Minor')
    Min_accuracy_Min=Min_accuracy_Min + 1;
    Min_out_ave=Min_out_ave+out;
    N = N+1;
end
[chord_class_string,out] = Full_Analysis_2(GsMin);
if strcmp(chord_class_string,'G# Minor')
    Min_accuracy_Min=Min_accuracy_Min + 1;
    Min_out_ave=Min_out_ave+out;
    N = N+1;
end
[chord_class_string,out] = Full_Analysis_2(AMin);
if strcmp(chord_class_string,'A Minor')
    Min_accuracy_Min=Min_accuracy_Min + 1;
    Min_out_ave=Min_out_ave+out;
    N = N+1;
end
[chord_class_string,out] = Full_Analysis_2(AsMin);
if strcmp(chord_class_string,'A# Minor')
    Min_accuracy_Min=Min_accuracy_Min + 1;
    Min_out_ave=Min_out_ave+out;
    N = N+1;
end
[chord_class_string,out] = Full_Analysis_2(BMin);
if strcmp(chord_class_string,'B Minor')
    Min_accuracy_Min=Min_accuracy_Min + 1;
    Min_out_ave=Min_out_ave+out;
    N = N+1;
end
[chord_class_string,out] = Full_Analysis_2(CAug);
if strcmp(chord_class_string,'C Augmented')
    Aug_accuracy_Aug=Aug_accuracy_Aug + 1;
    Aug_out_ave=Aug_out_ave+out;
    O = O+1;
end
[chord_class_string,out] = Full_Analysis_2(CsAug);
if strcmp(chord_class_string,'C# Augmented')
    Aug_accuracy_Aug=Aug_accuracy_Aug + 1;
    Aug_out_ave=Aug_out_ave+out;
    O = O+1;
end
[chord_class_string,out] = Full_Analysis_2(DAug);
if strcmp(chord_class_string,'D Augmented')
    Aug_accuracy_Aug=Aug_accuracy_Aug + 1;
    Aug_out_ave=Aug_out_ave+out;
    O = O+1;
```

```
end
[chord_class_string,out] = Full_Analysis_2(DsAug);
if strcmp(chord_class_string,'D# Augmented')
    Aug_accuracy_Aug=Aug_accuracy_Aug + 1;
    Aug_out_ave=Aug_out_ave+out;
    O = O+1;
end
[chord_class_string,out] = Full_Analysis_2(EAug);
if strcmp(chord_class_string,'E Augmented')
    Aug_accuracy_Aug=Aug_accuracy_Aug + 1;
    Aug_out_ave=Aug_out_ave+out;
    O = O+1;
end
[chord_class_string,out] = Full_Analysis_2(FAug);
if strcmp(chord_class_string,'F Augmented')
    Aug_accuracy_Aug=Aug_accuracy_Aug + 1;
    Aug_out_ave=Aug_out_ave+out;
    O = O+1;
end
[chord_class_string,out] = Full_Analysis_2(FsAug);
if strcmp(chord_class_string,'F# Augmented')
    Aug_accuracy_Aug=Aug_accuracy_Aug + 1;
    Aug_out_ave=Aug_out_ave+out;
    O = O+1;
end
[chord_class_string,out] = Full_Analysis_2(GAug);
if strcmp(chord_class_string,'G Augmented')
    Aug_accuracy_Aug=Aug_accuracy_Aug + 1;
    Aug_out_ave=Aug_out_ave+out;
    O = O+1;
end
[chord_class_string,out] = Full_Analysis_2(GsAug);
if strcmp(chord_class_string,'G# Augmented')
    Aug_accuracy_Aug=Aug_accuracy_Aug + 1;
    Aug_out_ave=Aug_out_ave+out;
    O = O+1;
end
[chord_class_string,out] = Full_Analysis_2(AAug);
if strcmp(chord_class_string,'A Augmented')
    Aug_accuracy_Aug=Aug_accuracy_Aug + 1;
    Aug_out_ave=Aug_out_ave+out;
    O = O+1;
end
[chord_class_string,out] = Full_Analysis_2(AsAug);
if strcmp(chord_class_string,'A# Augmented')
    Aug_accuracy_Aug=Aug_accuracy_Aug + 1;
    Aug_out_ave=Aug_out_ave+out;
    O = O+1;
end
[chord_class_string,out] = Full_Analysis_2(BAug);
if strcmp(chord_class_string,'B Augmented')
    Aug_accuracy_Aug=Aug_accuracy_Aug + 1;
    Aug_out_ave=Aug_out_ave+out;
    O = O+1;
end
[chord_class_string,out] = Full_Analysis_2(CDim);
if strcmp(chord_class_string,'C Diminished')
    Dim_accuracy_Dim=Dim_accuracy_Dim + 1;
    Dim_out_ave=Dim_out_ave+out;
    P = P+1;
end
```

```
[chord_class_string,out] = Full_Analysis_2(CsDim);
if strcmp(chord_class_string,'C# Diminished')
    Dim_accuracy_Dim=Dim_accuracy_Dim + 1;
    Dim_out_ave=Dim_out_ave+out;
    P = P+1;
end
[chord_class_string,out] = Full_Analysis_2(DDim);
if strcmp(chord_class_string,'D Diminished')
    Dim_accuracy_Dim=Dim_accuracy_Dim + 1;
    Dim_out_ave=Dim_out_ave+out;
    P = P+1;
end
[chord_class_string,out] = Full_Analysis_2(DsDim);
if strcmp(chord_class_string,'D# Diminished')
    Dim_accuracy_Dim=Dim_accuracy_Dim + 1;
    Dim_out_ave=Dim_out_ave+out;
    P = P+1;
end
[chord_class_string,out] = Full_Analysis_2(EDim);
if strcmp(chord_class_string,'E Diminished')
    Dim_accuracy_Dim=Dim_accuracy_Dim + 1;
    Dim_out_ave=Dim_out_ave+out;
    P = P+1;
end
[chord_class_string,out] = Full_Analysis_2(FDim);
if strcmp(chord_class_string,'F Diminished')
    Dim_accuracy_Dim=Dim_accuracy_Dim + 1;
    Dim_out_ave=Dim_out_ave+out;
    P = P+1;
end
[chord_class_string,out] = Full_Analysis_2(FsDim);
if strcmp(chord_class_string,'F# Diminished')
    Dim_accuracy_Dim=Dim_accuracy_Dim + 1;
    Dim_out_ave=Dim_out_ave+out;
    P = P+1;
end
[chord_class_string,out] = Full_Analysis_2(GDim);
if strcmp(chord_class_string,'G Diminished')
    Dim_accuracy_Dim=Dim_accuracy_Dim + 1;
    Dim_out_ave=Dim_out_ave+out;
    P = P+1;
end
[chord_class_string,out] = Full_Analysis_2(GsDim);
if strcmp(chord_class_string,'G# Diminished')
    Dim_accuracy_Dim=Dim_accuracy_Dim + 1;
    Dim_out_ave=Dim_out_ave+out;
    P = P+1;
end
[chord_class_string,out] = Full_Analysis_2(ADim);
if strcmp(chord_class_string,'A Diminished')
    Dim_accuracy_Dim=Dim_accuracy_Dim + 1;
    Dim_out_ave=Dim_out_ave+out;
    P = P+1;
end
[chord_class_string,out] = Full_Analysis_2(AsDim);
if strcmp(chord_class_string,'A# Diminished')
    Dim_accuracy_Dim=Dim_accuracy_Dim + 1;
    Dim_out_ave=Dim_out_ave+out;
    P = P+1;
end
[chord_class_string,out] = Full_Analysis_2(BDim);
```

```
if strcmp(chord_class_string,'B Diminished')
    Dim_accuracy_Dim=Dim_accuracy_Dim + 1;
    Dim_out_ave=Dim_out_ave+out;
    P = P+1;
end


Maj_ave=Maj_out_ave/M
Maj_acc=((Maj_accuracy_maj)/12)*100
Min_ave=Min_out_ave/N
Min_acc=((Min_accuracy_Min)/12)*100
Aug_ave=Aug_out_ave/O
Aug_acc=((Aug_accuracy_Aug)/12)*100
Dim_ave=Dim_out_ave/P
Dim_acc=((Dim_accuracy_Dim)/12)*100

end
```

MLA algorithm implemented in MATALB. Written by Anton Gouws

```
%This function impliments the Maximum Liklihood Algorhithm that is the
%backbone to the transcription process.

function [chord_class_string,out] = Full_Analysis_2(chord_to_test)
%Imports the data to compare to theory
chord_to_test_a=transpose(chord_to_test);
%Matrices and vectors containing the chords based on theory (Binary values
%used)
Aug =
[1,0,0,0,1,0,0,0,1,0,0,0;0,1,0,0,0,1,0,0,0,1,0,0;0,0,1,0,0,0,1,0,0,0,1,0;0,
0,0,1,0,0,0,1,0,0,0,1;1,0,0,0,1,0,0,0,1,0,0,0;0,1,0,0,0,1,0,0,0,1,0,0;0,0,1
,0,0,0,1,0,0,0,1,0;0,0,0,1,0,0,0,1,0,0,0,1;1,0,0,0,1,0,0,0,1,0,0,0;0,1,0,0,
0,1,0,0,0,1,0,0;0,0,1,0,0,0,1,0,0,0,1,0;0,0,0,1,0,0,0,1,0,0,0,1;];
Dim =
[1,0,0,1,0,0,1,0,0,0,0,0;0,1,0,0,1,0,0,1,0,0,0,0;0,0,1,0,0,1,0,0,1,0,0,0;0,
0,0,1,0,0,1,0,0,1,0,0;0,0,0,0,1,0,0,1,0,0,1,0;0,0,0,0,0,1,0,0,1,0,0,1;1,0,0
,0,0,0,1,0,0,1,0,0;0,1,0,0,0,0,0,1,0,0,1,0;0,0,1,0,0,0,0,0,1,0,0,1;1,0,0,1,
0,0,0,0,0,1,0,0;0,1,0,0,1,0,0,0,0,0,1,0;0,0,1,0,0,1,0,0,0,0,0,1;];
Maj =
[1,0,0,0,1,0,0,1,0,0,0,0;0,1,0,0,0,1,0,0,1,0,0,0;0,0,1,0,0,0,1,0,0,1,0,0;0,
0,0,1,0,0,0,1,0,0,1,0;0,0,0,0,1,0,0,0,1,0,0,1;1,0,0,0,0,1,0,0,0,1,0,0;0,1,0
,0,0,0,1,0,0,0,1,0;0,0,1,0,0,0,0,1,0,0,0,1;1,0,0,1,0,0,0,0,1,0,0,0;0,1,0,0,
1,0,0,0,0,1,0,0;0,0,1,0,0,1,0,0,0,0,1,0;0,0,0,1,0,0,1,0,0,0,0,1;];
Min =
[1,0,0,1,0,0,0,1,0,0,0,0;0,1,0,0,1,0,0,0,1,0,0,0;0,0,1,0,0,1,0,0,0,1,0,0;0,
0,0,1,0,0,1,0,0,0,1,0;0,0,0,0,1,0,0,1,0,0,0,1;1,0,0,0,0,1,0,0,1,0,0,0;0,1,0
,0,0,0,1,0,0,1,0,0;0,0,1,0,0,0,0,1,0,0,1,0;0,0,0,1,0,0,0,0,1,0,0,1;1,0,0,0,
1,0,0,0,0,1,0,0;0,1,0,0,0,1,0,0,0,0,1,0;0,0,1,0,0,0,1,0,0,0,0,1;];
CMaj=zeros(1,12);CsMaj=zeros(1,12);DMaj=zeros(1,12);DsMaj=zeros(1,12);EMaj=
zeros(1,12);FMaj=zeros(1,12);FsMaj=zeros(1,12);GMaj=zeros(1,12);GsMaj=zeros
(1,12);AMaj=zeros(1,12);AsMaj=zeros(1,12);BMaj=zeros(1,12);
CMin=zeros(1,12);CsMin=zeros(1,12);DMin=zeros(1,12);DsMin=zeros(1,12);EMin=
zeros(1,12);FMin=zeros(1,12);FsMin=zeros(1,12);GMin=zeros(1,12);GsMin=zeros
(1,12);AMin=zeros(1,12);AsMin=zeros(1,12);BMin=zeros(1,12);
CAug=zeros(1,12);CsAug=zeros(1,12);DAug=zeros(1,12);DsAug=zeros(1,12);EAug=
zeros(1,12);FAug=zeros(1,12);FsAug=zeros(1,12);GAug=zeros(1,12);GsAug=zeros
(1,12);AAug=zeros(1,12);AsAug=zeros(1,12);BAug=zeros(1,12);
CDim=zeros(1,12);CsDim=zeros(1,12);DDim=zeros(1,12);DsDim=zeros(1,12);EDim=
zeros(1,12);FDim=zeros(1,12);FsDim=zeros(1,12);GDim=zeros(1,12);GsDim=zeros
(1,12);ADim=zeros(1,12);AsDim=zeros(1,12);BDim=zeros(1,12);
```

```
%Assigns an individual vector for each chord such that matrix
%multiplication can be used in the MLA
for i = 1:12
    CAug(1,i)=Aug(1,i);
    CsAug(1,i)=Aug(2,i);
    DAug(1,i)=Aug(3,i);
    DsAug(1,i)=Aug(4,i);
    EAug(1,i)=Aug(5,i);
    FAug(1,i)=Aug(6,i);
    FsAug(1,i)=Aug(7,i);
    GAug(1,i)=Aug(8,i);
    GsAug(1,i)=Aug(9,i);
    AAug(1,i)=Aug(10,i);
    AsAug(1,i)=Aug(11,i);
    BAug(1,i)=Aug(12,i);
end
for i = 1:12
    CDim(1,i)=Dim(1,i);
    CsDim(1,i)=Dim(2,i);
    DDim(1,i)=Dim(3,i);
    DsDim(1,i)=Dim(4,i);
    EDim(1,i)=Dim(5,i);
    FDim(1,i)=Dim(6,i);
    FsDim(1,i)=Dim(7,i);
    GDim(1,i)=Dim(8,i);
    GsDim(1,i)=Dim(9,i);
    ADim(1,i)=Dim(10,i);
    AsDim(1,i)=Dim(11,i);
    BDim(1,i)=Dim(12,i);
end
for i = 1:12
    CMaj(1,i)=Maj(1,i);
    CsMaj(1,i)=Maj(2,i);
    DMaj(1,i)=Maj(3,i);
    DsMaj(1,i)=Maj(4,i);
    EMaj(1,i)=Maj(5,i);
    FMaj(1,i)=Maj(6,i);
    FsMaj(1,i)=Maj(7,i);
    GMaj(1,i)=Maj(8,i);
    GsMaj(1,i)=Maj(9,i);
    AMaj(1,i)=Maj(10,i);
    AsMaj(1,i)=Maj(11,i);
    BMaj(1,i)=Maj(12,i);
end
for i = 1:12
    CMin(1,i)=Min(1,i);
    CsMin(1,i)=Min(2,i);
    DMin(1,i)=Min(3,i);
    DsMin(1,i)=Min(4,i);
    EMin(1,i)=Min(5,i);
    FMin(1,i)=Min(6,i);
    FsMin(1,i)=Min(7,i);
    GMin(1,i)=Min(8,i);
    GsMin(1,i)=Min(9,i);
    AMin(1,i)=Min(10,i);
    AsMin(1,i)=Min(11,i);
    BMin(1,i)=Min(12,i);
end
%Multiplies the test vector with the chord database
m1=CMin*chord_to_test_a;
m2=CsMin*chord_to_test_a;
```

```
m3=DMin*chord_to_test_a;
m4=DsMin*chord_to_test_a;
m5=EMin*chord_to_test_a;
m6=FMin*chord_to_test_a;
m7=FsMin*chord_to_test_a;
m8=GMin*chord_to_test_a;
m9=GsMin*chord_to_test_a;
m10=AMin*chord_to_test_a;
m11=AsMin*chord_to_test_a;
m12=BMin*chord_to_test_a;
n1=CMaj*chord_to_test_a;
n2=CsMaj*chord_to_test_a;
n3=DMaj*chord_to_test_a;
n4=DsMaj*chord_to_test_a;
n5=EMaj*chord_to_test_a;
n6=FMaj*chord_to_test_a;
n7=FsMaj*chord_to_test_a;
n8=GMaj*chord_to_test_a;
n9=GsMaj*chord_to_test_a;
n10=AMaj*chord_to_test_a;
n11=AsMaj*chord_to_test_a;
n12=BMaj*chord_to_test_a;
o1=CAug*chord_to_test_a;
o2=CsAug*chord_to_test_a;
o3=DAug*chord_to_test_a;
o4=DsAug*chord_to_test_a;
o5=EAug*chord_to_test_a;
o6=FAug*chord_to_test_a;
o7=FsAug*chord_to_test_a;
o8=GAug*chord_to_test_a;
o9=GsAug*chord_to_test_a;
o10=AAug*chord_to_test_a;
o11=AsAug*chord_to_test_a;
o12=BAug*chord_to_test_a;
p1=CDim*chord_to_test_a;
p2=CsDim*chord_to_test_a;
p3=DDim*chord_to_test_a;
p4=DsDim*chord_to_test_a;
p5=EDim*chord_to_test_a;
p6=FDim*chord_to_test_a;
p7=FsDim*chord_to_test_a;
p8=GDim*chord_to_test_a;
p9=GsDim*chord_to_test_a;
p10=ADim*chord_to_test_a;
p11=AsDim*chord_to_test_a;
p12=BDim*chord_to_test_a;
%Chooses the maximum correlation number out of these variables
n=max([n1,n2,n3,n4,n5,n6,n7,n8,n9,n10,n11,n12,m1,m2,m3,m4,m5,m6,m7,m8,m9,m1
0,m11,m12,o1,o2,o3,o4,o5,o6,o7,o8,o9,o10,o11,o12,p1,p2,p3,p4,p5,p6,p7,p8,p9
,p10,p11,p12]);
%Returns the best guess chord, and the correlation value
if n==o1
    disp('Best Guess Chord is C augmented');
    out=o1;
    chord_class_string='C Augmented';
elseif n==o2
    disp('Best Guess Chord is C# augmented');
    out=o2;
    chord_class_string='C# Augmented';
elseif n==o3
    disp('Best Guess Chord is D augmented');
```

```
    out=o3;
    chord_class_string='D Augmented';
elseif n==o4
    disp('Best Guess Chord is D# augmented');
    out=o4;
    chord_class_string='D# Augmented';
elseif n==o5
    disp('Best Guess Chord is E augmented');
    out=o5;
    chord_class_string='E Augmented';
elseif n==o6
    disp('Best Guess Chord is F augmented');
    out=o6;
    chord_class_string='F Augmented';
elseif n==o7
    disp('Best Guess Chord is F# augmented');
    out=o7;
    chord_class_string='F# Augmented';
elseif n==o8
    disp('Best Guess Chord is G augmented');
    out=o8;
    chord_class_string='G Augmented';
elseif n==o9
    disp('Best Guess Chord is G# augmented');
    out=o9;
    chord_class_string='G# Augmented';
elseif n==o10
    disp('Best Guess Chord is A augmented');
    out=o10;
    chord_class_string='A Augmented';
elseif n==o11
    disp('Best Guess Chord is A# augmented');
    out=o11;
    chord_class_string='A# Augmented';
elseif n==o12
    disp('Best Guess Chord is B augmented');
    out=o12;
    chord_class_string='B Augmented';
elseif n==p1
    disp('Best Guess Chord is C diminished');
    out=p1;
    chord_class_string='C Diminished';
elseif n==p2
    disp('Best Guess Chord is C# diminished');
    out=p2;
    chord_class_string='C# Diminished';
elseif n==p3
    disp('Best Guess Chord is D diminished');
    out=p3;
    chord_class_string='D Diminished';
elseif n==p4
    disp('Best Guess Chord is D# diminished');
    out=p4;
    chord_class_string='D# Diminished';
elseif n==p5
    disp('Best Guess Chord is E diminished');
    out=p5;
    chord_class_string='E Diminished';
elseif n==p6
    disp('Best Guess Chord is F diminished');
    out=p7;
```

```
    chord_class_string='F Diminished';
elseif n==p7
    disp('Best Guess Chord is F# diminished');
    out=p7;
    chord_class_string='F# Diminished';
elseif n==p8
    disp('Best Guess Chord is G diminished');
    out=p8;
    chord_class_string='G Diminished';
elseif n==p9
    disp('Best Guess Chord is G# diminished');
    out=p9;
    chord_class_string='G# Diminished';
elseif n==p10
    disp('Best Guess Chord is A diminished');
    out=p10;
    chord_class_string='A Diminished';
elseif n==p11
    disp('Best Guess Chord is A# diminished');
    out=p11;
    chord_class_string='A# Diminished';
elseif n==p12
    disp('Best Guess Chord is B diminished');
    out=p12;
    chord_class_string='B Diminished';
elseif n==m1
    disp('Best Guess Chord is C minor');
    out=m1;
    chord_class_string='C Minor';
elseif n==m2
    disp('Best Guess Chord is C# Minor');
    out=m2;
    chord_class_string='C# Minor';
elseif n==m3
    disp('Best Guess Chord is D Minor');
    out=m3;
    chord_class_string='D Minor';
elseif n==m4
    disp('Best Guess Chord is D# Minor');
    out=m4;
    chord_class_string='D# Minor';
elseif n==m5
    disp('Best Guess Chord is E Minor');
    out=m5;
    chord_class_string='E Minor';
elseif n==m6
    disp('Best Guess Chord is F Minor');
    out=m6;
    chord_class_string='F Minor';
elseif n==m7
    disp('Best Guess Chord is F# Minor');
    out=m7;
    chord_class_string='F# Minor';
elseif n==m8
    disp('Best Guess Chord is G Minor');
    out=m8;
    chord_class_string='G Minor';
elseif n==m9
    disp('Best Guess Chord is G# Minor');
    out=m9;
    chord_class_string='G# Minor';
```

```
elseif n==m10
    disp('Best Guess Chord is A Minor');
    out=m10;
    chord_class_string='A Minor';
elseif n==m11
    disp('Best Guess Chord is A# Minor');
    out=m11;
    chord_class_string='A# Minor';
elseif n==m12
    disp('Best Guess Chord is B Minor');
    out=m12;
    chord_class_string='B Minor';
elseif n==n1
    disp('Best Guess Chord is C major');
    out=n1;
    chord_class_string='C Major';
elseif n==n2
    disp('Best Guess Chord is C# major');
    out=n2;
    chord_class_string='C# Major';
elseif n==n3
    disp('Best Guess Chord is D major');
    out=n3;
    chord_class_string='D Major';
elseif n==n4
    disp('Best Guess Chord is D# major');
    out=n4;
    chord_class_string='D# Major';
elseif n==n5
    disp('Best Guess Chord is E major');
    out=n5;
    chord_class_string='E Major';
elseif n==n6
    disp('Best Guess Chord is F major');
    out=n6;
    chord_class_string='F Major';
elseif n==n7
    disp('Best Guess Chord is F# major');
    out=n7;
    chord_class_string='F# Major';
elseif n==n8
    disp('Best Guess Chord is G major');
    out=n8;
    chord_class_string='G Major';
elseif n==n9
    disp('Best Guess Chord is G# major');
    out=n9;
    chord_class_string='G# Major';
elseif n==n10
    disp('Best Guess Chord is A major');
    out=n10;
    chord_class_string='A Major';
elseif n==n11
    disp('Best Guess Chord is A# major');
    out=n11;
    chord_class_string='A# Major';
elseif n==n12
    disp('Best Guess Chord is B major');
    out=n12;
    chord_class_string='B Major';
```

```
end
end
```

Program that generates sine wave chords to confirm the 'mirchromagram.m' function, including upper partial portion added.

```
f1=523.251;    %frequency corresponding to C
f2=659.255;    %frequency corresponding to E
f3=783.991;    %frequency corresponding to G
co1=2*pi*f1;   %fundamental frequency coefficients
co2=2*pi*f2;
co3=2*pi*f3;
co1_upper1=2*pi*(2*f1);    %first upper partials
co2_upper1=2*pi*(2*f2);
co3_upper1=2*pi*(2*f3);
co1_upper2=2*pi*(3*f1);    %second upper partials
co2_upper2=2*pi*(3*f2);
co3_upper2=2*pi*(3*f3);
sig1=0.2*sin(co1*x);       %fundamental frequencies
sig2=0.2*sin(co2*x);
sig3=0.2*sin(co3*x);
sig4=0.1*sin(co1_upper1*x); %first upper partial with scaling factor
sig5=0.05*sin(co2_upper1*x);
sig6=0.05*sin(co3_upper1*x);
sig7=0.012*sin(co1_upper2*x);   %second upper partial with scaling factor
sig8=0.008*sin(co2_upper2*x);
sig9=0.005*sin(co3_upper2*x);
sig=sig1+sig2+sig3+sig4+sig5+sig6+sig7+sig8+sig9;
```