Analysis and resynthesis of polyphonic music

Douglas John Edgar Nunn

Analysis and Resynthesis of Polyphonic Music

Douglas John Edgar Nunn

A Dissertation Submitted to the Faculty of Science
in Candidacy for the Degree of Doctor of Philosophy

School of Engineering
University of Durham

1997

DOUGLAS NUNN

ANALYSIS AND RESYNTHESIS OF POLYPHONIC MUSIC

ABSTRACT

This thesis examines applications of Digital Signal Processing to the analysis, transformation, and resynthesis of musical audio. First I give an overview of the human perception of music. I then examine in detail the requirements for a system that can analyse, transcribe, process, and resynthesise monaural polyphonic music. I then describe and compare the possible hardware and software platforms. After this I describe a prototype hybrid system that attempts to carry out these tasks using a method based on additive synthesis. Next I present results from its application to a variety of musical examples, and critically assess its performance and limitations. I then address these issues in the design of a second system based on Gabor wavelets. I conclude by summarising the research and outlining suggestions for future developments.

ANALYSIS AND RESYNTHESIS OF POLYPHONIC MUSIC

DOUGLAS JOHN EDGAR NUNN

A DISSERTATION SUBMITTED TO THE FACULTY OF SCIENCE
IN CANDIDACY FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

SCHOOL OF ENGINEERING
UNIVERSITY OF DURHAM

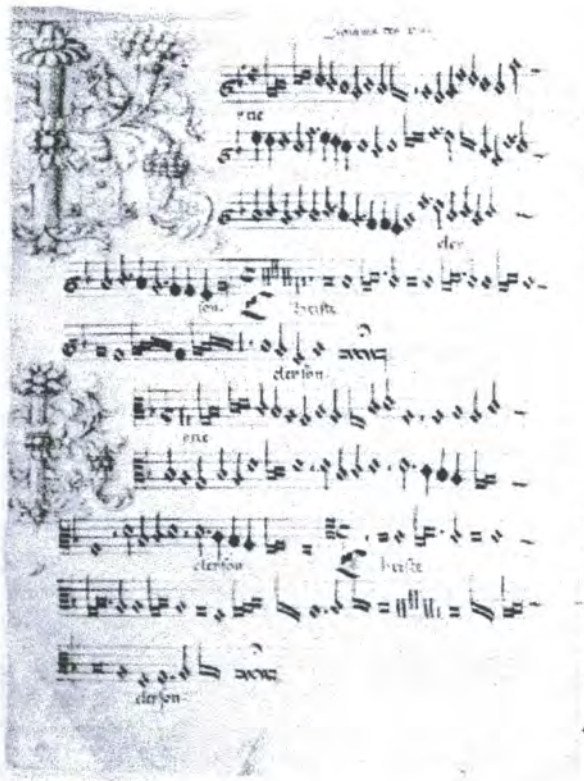
1997

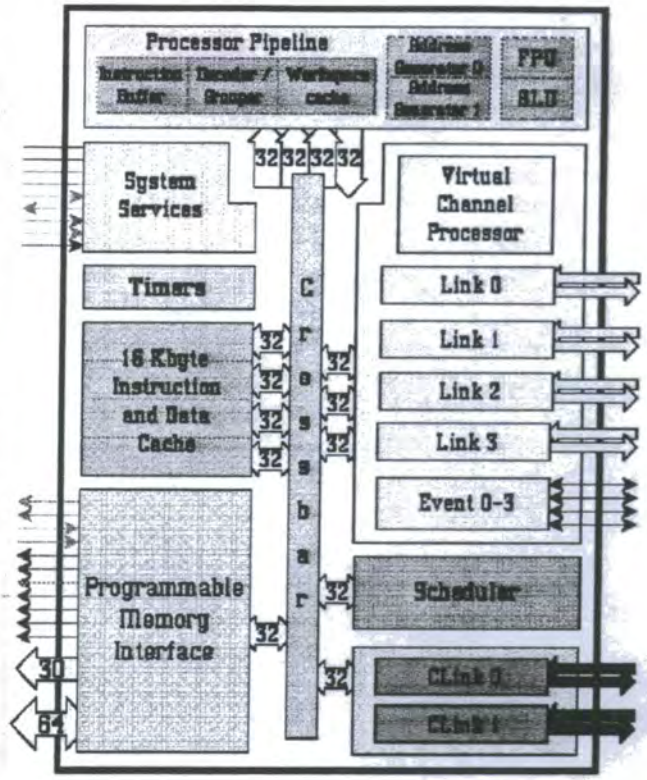
0. Chapter Zero

0.1 Table of contents

0.2 Table of figures

0.3 List of tables





0.4 Declaration

None of this work has previously been submitted for a degree in this or any other university.

1. Introduction

What is music? How do we perceive it? How do we separate simultaneous sounds? How do we mentally represent timbre? How do we form higher concepts such as harmony, melody, and rhythm? How could a computer mimic the perception process? How could it write down the score? How could we fit more music onto a CD? How could a computer explore the vast multidimensional space of possible timbres? How can we transform musical sounds? This work develops a framework for computer analysis and resynthesis of polyphonic music in which these questions are addressed.

1.1 Background

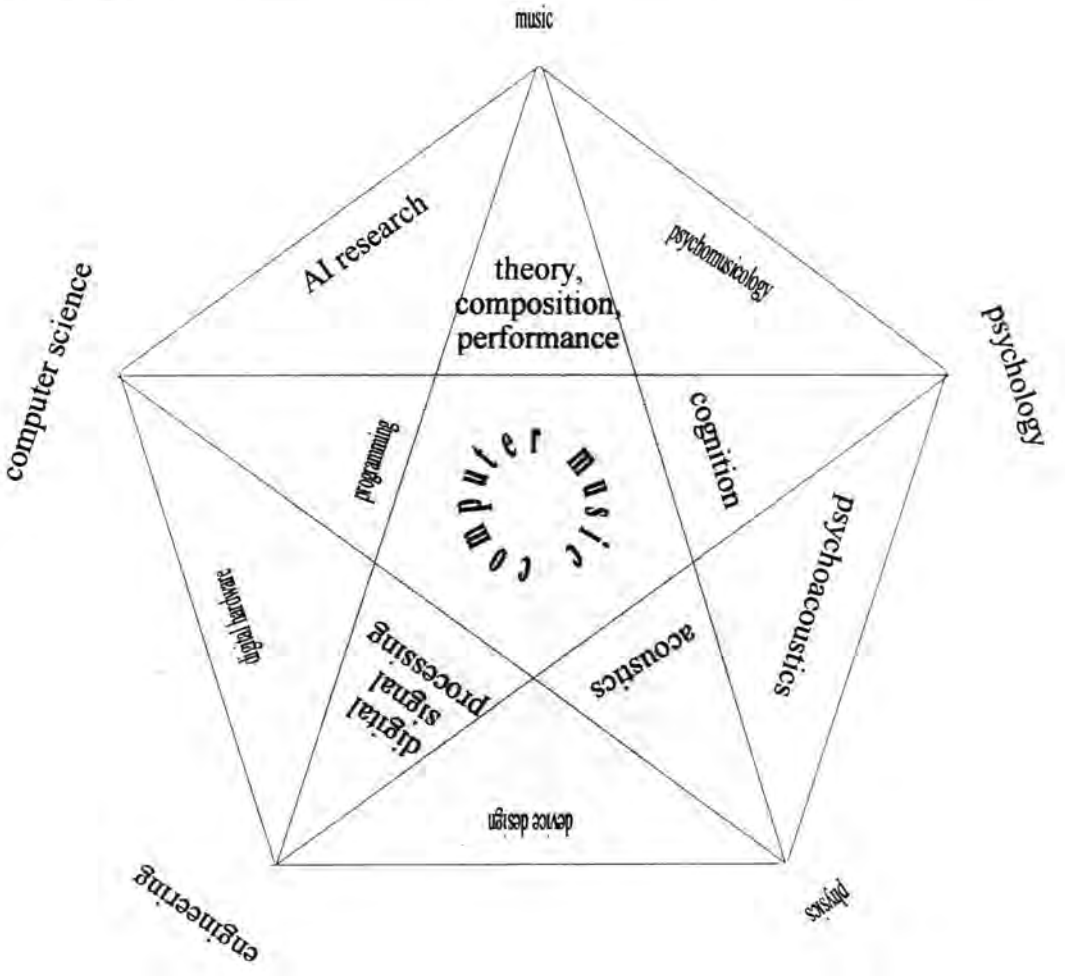


Figure 1 - Disciplinary context of computer music.

The disciplinary context of computer music is shown in Figure 1, after Moore.[MooreF 90] The shaded area represents the fields with which this thesis will be concerned.

The needs of electroacoustic[1] composition are complex and varied. Composers often wish to use entirely new sounds, and much research has concentrated on the development and exposition of new synthesis techniques. Yet most current techniques only allow the creation of a different but limited subset of timbres, and hence the ability to explore another small part of what is loosely called timbre space. Composers also wish to use sounds that resemble acoustic instruments. This has led to much research into *analysis-by-synthesis*, an attempt to emulate the sound of individual acoustic instruments by adjusting synthesis parameters. However, this cannot be guaranteed to reproduce every nuance of a sound, as even the most realistic physical models make approximations. Thus, while we can make a synthetic violin that is 'good enough' to use, we still cannot reproduce the sound of a specific Stradivarius or mimic the interpretation of a real violinist. Composers often want to incorporate existing sounds into their own compositions, but this is generally difficult or impossible, except for directly sampling a wave file, which does not permit musical editing. One goal of this research, then, is to achieve *synthesis-by-analysis*, whereby an arbitrary sound can be described and reproduced perfectly, as this would allow the use of any existing sound source, completing the loop between synthesis and analysis. While this is possible for monophonic sounds, the quest for *polyphonic* synthesis-by-analysis has so far eluded solution.

We also have an interest in examining our own cognition of music. Here, analysis of audio is more important than synthesis. Important questions remain unsolved concerning how we perceive and mentally represent music. While simple single-tone phenomena have been researched, our perception of timbre and timbral nuance is not well understood. Less still is known about perception of chords, melodies, and other musical constructs, let alone more abstract and personal questions of aesthetics and style. Musicologists and educationalists also wish to be able to analyse performance practice; that which distinguishes a performance by a competent instrumentalist from one by a poor instrumentalist. In both of these cases, too, the greatest difficulty lies in untangling the polyphony.

This thesis examines how musical information can be derived from an *arbitrary* raw waveform, and concentrates on methods by which note *and* timbral information can be deduced, in order to allow display and transformation of musically useful parameters, while retaining the generality required for resynthesis. This will not only allow practical tools for analysis, transformation, and resynthesis, but may offer insights into our own perception of music.

---

[1] I shall mention at this early point my dislike of the term 'electroacoustic music'. All music is acoustic. However, I will use it in its sense of 'music featuring electr(on)ic instruments including computers'.

1.2 Outline of this thesis

Chapter 2 examines human music perception in detail. In chapter 3, the motivation for computer analysis and resynthesis of audio is discussed, and applications are outlined. This is followed by a discussion of the computational requirements and possible architectures in chapter 4. Earlier research on the topic is reviewed in chapter 5. Chapter 6 presents a model based on multirate additive synthesis that attempts to recognise and characterise musical events. Chapter 7 evaluates its analysis performance on a variety of musical examples, and discusses its limitations. Chapter 8 addresses the issues raised by redesigning the system using a related technique based on Gabor wavelets, which appears to address many of the problems. Finally, conclusions are drawn and proposals for future work are presented in chapter 9.

2. Music perception

In this chapter I first examine whether and how music can be defined. Next I give an overview of the human auditory system and its functioning. I then discuss the process of source separation. Next I outline the physical and psychoacoustical properties of notes in a musical context, and discuss possibilities for a computer model.

2.1 The nature of music

2.1.1 What is music?

Defining the word 'music' is only slightly less difficult than defining art. Indeed, the very question of whether music can be defined may prompt heated debate from those who choose to view music as inherently undefinable. However, as this work is entirely focused on music, I feel it important to endeavour to clarify what I mean by the word. Here I am *not* trying to distinguish good music from bad music, but trying to distinguish music from *non-music*.

Note also that throughout this work I am concentrating solely on the audio, separating it from other aspects of music such as the visuals of a performance. While there is some evidence that vision can affect audition (as illustrated by the McGurk effect[2]) and vice versa[Churchland, Cohen, Slaney 95b], this work will be examining music from the unimodal viewpoint of a *blind* listener.

In order to discuss the definitions, we will apply them to several audio waveforms. Many of them are extreme cases chosen specifically to illustrate the differences.

P1) *Yesterday*, written by John Lennon and Paul McCartney and performed by the Beatles on vocals, guitar, bass guitar, percussion, and string quartet.

P2) *Agony*, conceived by the author. This is deliberately unpleasant music, and features the sounds of pneumatic drills, chainsaws, dog fights, crying babies, and fingernails on a blackboard. It is played at 0 dB outside the diplomatic mission of one's choice.

P3) *4'33"*, written by John Cage. This piece consists of 273 seconds of silence, performed using a grand piano.

P4) *White Haze*, conceived by the author. This piece consists of 273 seconds of white noise, performed using a Geiger counter and a lump of uranium.

P5) *Symphony for Helicopter*, conceived by the author. This piece makes extensive use of the sampled sound of a helicopter, manipulated by an array of audio processing tools, and is performed via tape.

P6) *Scramble for Helicopter*, conceived by the author. This is the actual sound produced by a helicopter in its usual military context.

P7) *Winword*, conceived by the author. This is created by renaming WINWORD.EXE, the executable binary file for Microsoft Word for Windows, to WINWORD.SND.

P8) *No Rain*, conceived by the author. This is the sound made by the windscreen wipers of a car on a dry day.

---

[2] The visual image of a speaker saying "ba" or "va" can affect whether an ambiguous sound is taken as one or the other syllable.

P9) *Dream*, conceived by Martin Luther King. This is a recording of MLK's "I Have a Dream" speech.

Below are six definitions we will examine:-

D1) Music is the language of sound.

D2) Music is any sound.

D3) Music is any sound intended as musical.

D4) Music is any sound perceived as musical.

D5) Music is any sound with regular patterns in time and frequency.

D6) Music is organised sound.

"Music is the language of sound." The analogy with languages is at times useful, but if music is a language, what meaning does it convey? A voice recognition system can detect the spoken phonemes 'K', 'A', 'T' as the symbolic "cat" and flash a small feline on the screen, at which point we deduce that it has *understood* the sound correctly. The path from phonemes to spelling to concept, from low-level to high-level, is relatively clear. However, for music, it is not known what the intermediate representations are. It is difficult to assign meaning to 'a high F# on a bassoon', and harder still to define the meaning of a whole piece of music. (An alternative definition is "Music is the art of sound", but this simply shifts us to an even larger problem of defining art, where one man's work of art is another man's pile of bricks.) This point is also raised by Wiggins, who points out that "If there is an analogy between the syntax of language and musical structure, what, if any, is the relationship between linguistic semantics and the 'meaning of music'? Indeed, it is by no means clear that such 'meaning' exists."[Wiggins] For this reason we ignore definition 1.

"Any sound is music." This would imply that white noise, the sound of a helicopter on its pad, and speech, are all musical. Most people would consider pure speech to *not* be music. Yet, if speech is not music, we should examine the continuum from pure speech to speech with background music to music heavily employing speech, and ask at which point it *becomes* music. This definition could exclude 4'33" if it were defined as the tape version on headphones, although the 'live' version would contain sounds such as footsteps and background concert hall noise.

"Music is any sound intended as music" is less broad, and would include the helicopter in a musical context, and 4'33". It would exclude the original helicopter, the white noise, and the speech. However, is the composer's assertion of its musicality a sufficient condition? Composers can use any available tool to create an audio waveform, and are not restricted to the vibrations of physical objects and electrical circuits. A composer would be free to assert that Winword is music, maybe adding in program notes that it represents "a para-metaphysical hyperconscious realisation of a pseudo-surrealistic sonic installation exploring the interconnectedness of meaning and substance, created by a novel and efficient

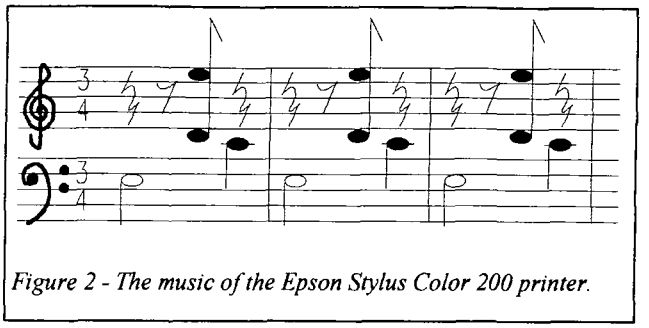


Figure 2 - The music of the Epson Stylus Color 200 printer.

computational paradigm". Most people would describe this "music" as not being very musical, in all probability indistinguishable from White Haze. In fact, this definition is not even a necessary condition as it excludes pleasant but non-human-inspired music such as the windscreen wipers or a squeaky gate hinge. The printer noise shown in Figure 2 is in a twelve-tone scale and a fairly regular tempo – the low E is the printing, the middle C is the print head returning, and the quavers in the treble clef are caused by the paper advance mechanism.

"Music is any sound perceived as musical." This is perhaps the best definition, as it includes the mechanically and unintentionally produced music described above. However, it is also a weak and circular definition.

"Music is any sound with regular patterns in time and frequency." This would appear to include the windscreen wipers but exclude 4'33". This is too inclusive and, like D2, cannot differentiate music from other audio, unless we further specified the types of 'regular patterns' we expected to see. It would certainly include the printer noise as it has a regular rhythm and a pitch structure.

"Music is organised sound" is ambiguous – it may be taken to mean either "sound organised by a composer" or "sound that has *self*-organisation" – in which case the definition can be considered equivalent to either "sound intended as music" (D3) or "sound with regular patterns" (D5).

This table summarises the distinctions between the definitions, ignoring definitions 1 and 6.

		D2 **any sound**	**D3** **intended as musical**	**D4** **perceived as musical**	**D5** **regular patterns**
P1	**Yesterday**	yes	yes	yes	yes
P2	**Agony**	yes	yes ?	yes ?	yes
P3	**4'33"**	no ?	yes	yes ?	no
P4	**White Haze**	yes	yes	no	no
P5	**Symphony**	yes	yes	yes	yes
P6	**Scramble**	yes	no	no	yes
P7	**Winword**	yes	yes ?	no	no ?
P8	**No Rain**	yes	no	yes ?	yes
P9	**Dream**	yes	no	no	yes

Table 1 - Applications of definitions of music to nine pieces.

Scientific research has allowed us to explain much of the world around us by breaking an object into its components, analysing them, and using their properties to infer the properties of the whole. Given that

we have an increasingly complete understanding of hailstorms, bridges, raspberries, holography, plate tectonics, malaria, prime numbers, and sea urchins, why do we apparently know so little about the phenomenon of music, which we may subject ourselves to for many hours per day?

One possible reason is the difficulty of describing our reactions to music in concise terms. It is not surprising that we have failed to concisely answer the question "what is jazz?" when we cannot answer the simpler question "what is the sound of a trombone?". A listener might describe it in terms of its similarity to the sound of a trumpet or a baritone, but this is no more useful than knowing that a lime tastes "a bit like a lemon".

Another reason that music has primarily been examined in the artistic rather than the scientific context is historical – the computational power required has not existed until now. However, recent leaps in processing power have brought many musical problems from the realm of the impossible to the feasible, and computers are certainly capable of mimicking some aspects of musical perception. It is worth noting the philosophical arguments against artificial intelligence – that a computer is fundamentally unable to carry out high-level tasks. A good example is computer chess; as chess programs reached higher levels of competence, they were set against increasingly skilled human players, and detractors claimed that it would still be impossible to beat the best human, citing factors such as emotion, intuition, and experience. Now that victories at the highest level have been achieved, there seems little doubt that the task of winning a chess game can be successfully carried out algorithmically. However, for AI researchers the larger question is "In what manner is the chess strategy similar to those we use?". We can describe how the computer is 'thinking' in hard program code, but it is ridiculous to suggest that Karpov is thinking in the same manner. For the task of listening to music, similar arguments apply. A program may be able to carry out tasks such as transcription, but this does not necessarily imply that it is hearing the music the way we hear it. One challenge, therefore, is to ensure that the analysis model reflects our own listening as far as is reasonable.

2.1.2 Time-frequency representations

Perhaps the only thing that can be said with complete certainty about music is that it is a continuous function of time – variations in air pressure. (Digital devices encode jumps in value, but the DAC output must be continuous.) If its first derivative is allowed to be discontinuous, i.e., the wave can contain corners, then we know[3] that its spectrum must be bounded by $O(1/f^2)$. This implies that its amplitude spectrum is bounded by a line falling by 6 dB/octave, and its power spectrum is bounded by a 12 dB/octave line.[Bracewell]

[3] As Bracewell concisely puts it, if the n^{th} derivative of a function becomes impulsive, then its spectrum is bounded by $O(f^{-n})$ as $f \to \infty$.

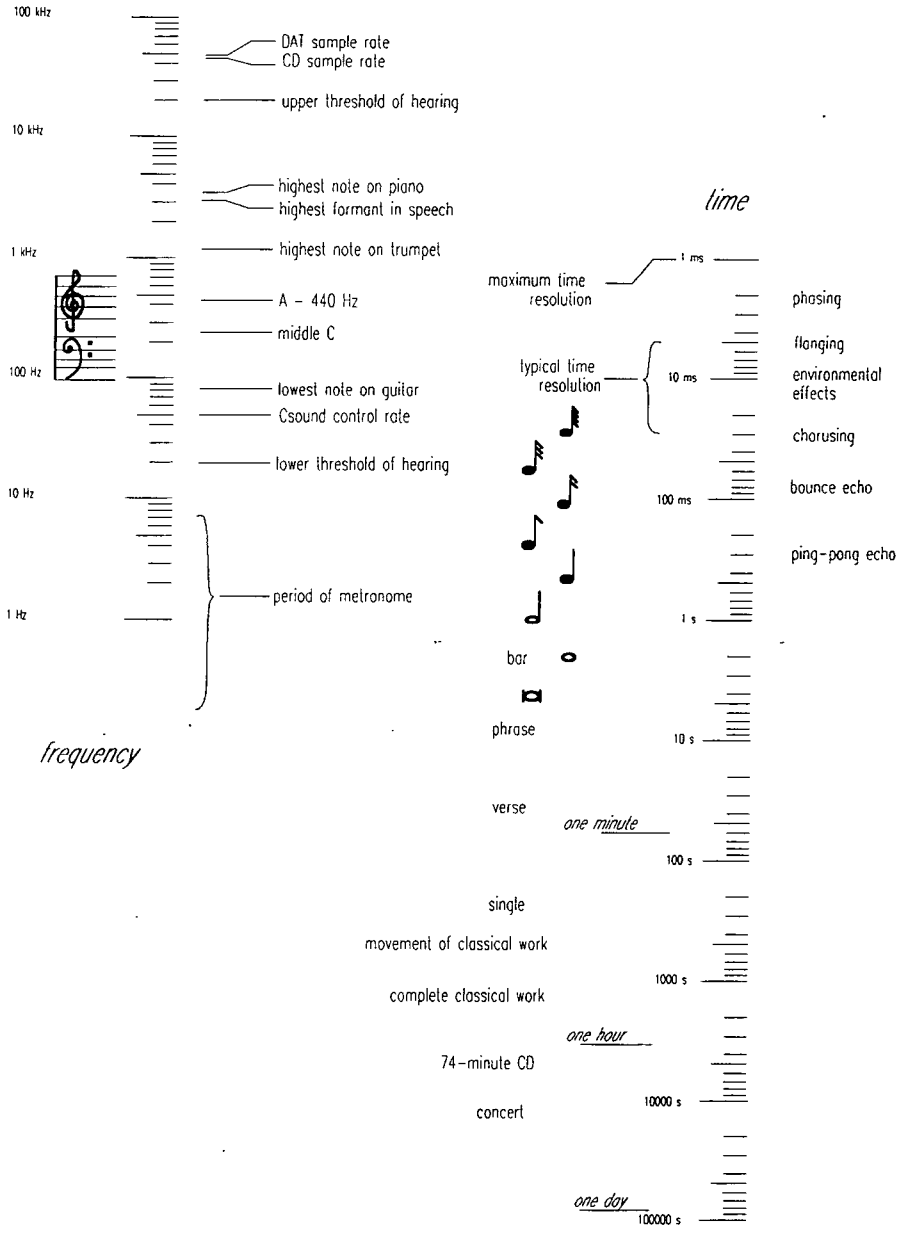


Figure 3 - The frequency-time axis.

There is a curious duality inherent in an audio waveform; we interpret it as both a time-domain and a frequency-domain representation, and treat seconds and Hertz as if they were independent dimensions. Figure 3 illustrates the frequency-time axis.

It has been shown using α-EEG (electroencephalogram) measurements that sounds above 20 kHz are sensed, and can affect our subjective perception of sound.[Oohashi 89, Oohashi 91, Oohashi 93] This was named the 'hypersonic effect', and was demonstrated using an Indonesian gamelan ensemble.[4] However, ultra-

[4] This is a good choice of source material – most gamelan instruments are percussive, so we would expect more high frequencies to be present.

high frequencies reaching us by non-aural paths raise the practical difficulties of designing recording and playback equipment. While conventionally recorded music may be missing something, it is clearly not missing much. It will thus be of little value to speculate on such details until we have a satisfactory explanation of the main part of the hearing range.

The conventional figure quoted for the top end of the range is around 20 kHz, although our high-frequency perception decreases with age (*presbycusis*). The sensitivity is highest in early childhood.[Rossing] In general, the power at these high frequencies is low. In this context, however, it is important to bear in mind that the notion of a limiting frequency is inaccurate; there is inevitably a finite roll-off that never reaches the conceptual minus infinity decibels. Also, while we can hear frequencies of 10-20 kHz, our resolution is measurably poorer than at lower frequencies.

The normal musical spectrum is from 20 kHz down to 20 Hz, a large range of ten octaves. Our perception of frequency ends at around 20 Hz. In musical practice, the lowest notes in the contrabass register are around 30 Hz. Very low pitches are the exception rather the rule, and are less musically useful.

The next frequency range is the transition region between low notes and fast rhythms. The only musical sounds with fundamentals in this range are rattles or drum rolls. (This frequency range is also used for effects such as vibrato, tremolo, and flutter-tonguing.) The sensation of low pitch blends into the sensation of a fast rhythm. Semiquavers at the dance music standard tempo of 120 bpm are at 8 Hz. This raises the question of whether rhythms can be said to interact with the bass, given that the 'harmonics' of the rhythm would fall at the same frequencies as the bass. If not, the alternative question is how do we dissociate information in the fast rhythm domain from that in the low pitch domain when both are functions of the same time axis?

As the frequency keeps falling, the units become seconds and minutes, not Hertz or milliHertz. The information is perceived in terms of beats, bars, phrases, sections, and so on. This is illustrated above – in the figure, many of the values shown are approximate – the note durations use a typical tempo of 120 crotchet beats per minute, and the figures on the right for stereo effects[Hall] are intermediate values in overlapping ranges.

2.1.3 The frequency domain – scales and temperament

The description above of frequency should be extended further. When dealing with the frequency domain, we actually treat it as if it were two dimensions, f and log(f). There is also now considerable support in the music cognition and psychoacoustic communities for log(f) itself being treated as multidimensional in order to recognise the 'closeness' of C to G.[Shephard, Moreno]

The harmonics of a single note are spaced approximately evenly on the linear frequency axis, (although frequencies can be stretched slightly, as discussed later). The pitches of the notes in the scale are related

according to their log-frequency. Thus, we can view relationships between frequencies on either a logarithmic or a linear scale. We effectively do both by choosing the base of logs appropriately. The most fundamental attribute of music world-wide is octave equivalence.[5] When a male and a female sing together, the consonance is greatest when the frequency ratio is 2:1 – an interval of an octave. Since the octave ratio is 2:1, the base 2 will come into the equations. The log to the base 2 of the ratio 3:2 is 0.5849, which happens to be very close to 7/12. The log of 5:4 is 0.3219, close to 4/12, and the log of 9:8 is 0.1699, close to 2/12. Thus, these intervals, and by extension all other intervals in the twelve-tone scale, can be interpreted in both the linear frequency domain and the domain of log-frequency to the base 1.05946..., the twelfth root of two.

It is informative to ask whether any other tuning systems are feasible in this way. The log of 1½, as mentioned above, is 0.5849. The only other small fractions[6] which are relatively close to this are 3/5 (0.6) and 4/7 (0.5714). The former would imply an octave with five equal subdivisions (corresponding to 2.4 semitones), and the interval in question is three of these divisions. Curiously enough, this does exist, and is known as the *slendro* scale.[Sundberg 91] It is only common in Indonesia (although it has also been found in Uganda), and even there is less used than the 12-note scale.[7] The second option would imply seven subdivisions (of 1.714 semitones), the 3:2 ratio being four of these. This scale has been reported in Thailand and Uganda.[Burns] These two alternative scales, while permitting the 3:2 ratio, do not give simple figures for the 5:4 or 6:5 ratios, and are thus less suited to the harmonic series.

We have thus shown by simple arithmetic the conclusion that 12 notes per octave is the most natural and pleasing, since the ratios 1:2:3:4:5:6:8:9 fit into it closely, an observation that is resoundingly backed up by actual musical practice world-wide. This also has led to interest from a group-theoretic point of view.[Balzano]

This method of making all the semitones equal is known as equal temperament. There are various alternatives, where some semitones are slightly larger than others. This permits certain intervals to be closer to the integer ratios, as the expense of making other intervals more out-of-tune. This allows the

---

[5] No rule is complete without its exceptions, but in this case such exceptions are hard to find. An alternative tuning system was devised by Bohlen and Pierce[Mathews 91, Moreno], where the fundamental interval was the ratio 3:1 rather than 2:1. This is consonant, but rather large. Two such 'tritaves' span the equivalent of three octaves plus one tone, a rather dissonant interval. The tritave is divided into 13 equal intervals, with a 9-note subset (0/1/3/4/6/7/9/10/12) used as the scale. Our overlearnt familiarity with octave relationships makes this particularly difficult to listen to in the manner intended.

[6] Ratios of larger integers are possible; of these, some are multiples of twelve, such as the Arab-Persian 24-note scale; others that have been used are 19-, 31-, and 41-note octaves. However, with such scales, the distinction between consecutive notes becomes much more difficult to hear.

[7] It has been said that in Indonesia, there are as many different scales as gamelans. This may largely be attributed to the fact that most *gamelan* instruments are gongs or bars which have an inharmonic partial structure and cannot be tuned easily.

intonation to be closer to 'just' intonation for some keys, but adds dissonance to music in other keys. In most contemporary music, equal temperament is adopted as the best compromise.

We have assumed that all of the instruments are in the same temperament. An important exception is in music created from many polyphonic sources. Rave and techno music are good examples – since it is crucial to align the temporal structure of several independent sources (e.g. two record decks), the pitches will be shifted by an amount that is probably not an exact number of semitones. It is not uncommon in this genre for there to be several temperaments at once.

Complicating the picture again, the instruments themselves can have an important effect on tuning. Consecutive strings of stringed instruments are separated by intervals of a fourth (double bass, guitar, bass guitar) or a fifth (violin, viola, cello), and the musician tunes them in just intonation. On valved brass instruments, each valve adds a constant amount to the length of tubing, rather than multiplying it by a constant. Thus, the combination of two or more valves adds slightly less than necessary, since $1+a+b < (1+a)(1+b)$.[8] In both these cases, the player has fine continuous control over the intonation, and can use this to correct mistuning or to add nuances. Even the octave equivalence can be distorted slightly; pianos are typically tuned slightly flat in the bass, and slightly sharp at the top end. This is known as 'Railsback stretching'. **[Wood, Schuck, Railsback, MartinD]**

Even when a fixed scale is in use, the fundamental pitches are not restricted to this discrete set. Notes are often played 'deliberately' out of tune – an instrument stands out more in the auditory scene if it is a little sharp. Notes might also be mistuned due to inaccuracy in controlling a continuous device (such as a trombone slide or a finger on a fretless string). The pitch of woodwind and brass instruments can be altered by the shape of the mouth. A high degree of expressiveness in music stems from pitch inflections such as glissando, portamento, and vibrato.

Despite these factors, the tuning is likely to be close to the equal-tempered scale. Thus, if we examine the distribution of frequencies in western music, we find that most of them are close to

$$A*(2^{M/12})*N$$

where A is a fixed tuning standard, M is an integral 'note number', and N is the harmonic. M and N need not be integers, but the point here is that they usually are very close, and A is usually fixed for each piece. The factor $\sqrt[12]{2}$ is what we know as a semitone – some musics use $\sqrt[5]{2}$, $\sqrt[7]{2}$, $\sqrt[19]{2}$, $\sqrt[24]{2}$, $\sqrt[37]{2}$, or $\sqrt[13]{3}$ as the basic pitch interval, but these are a tiny minority.

Generally, the octave relationship is exploited fully, with music staying in a fixed key, so the distribution is better described as

$$A*(2^{(K+M)/12})*N$$

---

[8] *Compensation* is used to correct this error on larger instruments.

where O is an integer, K is the nearly fixed key, and M is a pitch class from 0 to 11.

M may be further restricted to a subset of {0,1,2,3,4,5,6,7,8,9,10,11}. For a piece in a major key, we use {0,2,4,5,7,9,11}. The major pentatonic, which is widely used in Western[9] and Eastern music, uses {0,2,4,7,9}. F# major pentatonic includes the five black notes. For 7-note Western[10] diatonic scales, the notes are {0,2,4,5,7,9,11}, given by solutions to:-

$$-1 \leq (7x) \bmod 12 \leq 5$$

The Ionian (major), Dorian, Phrygian, Lydian, Mixolydian, Aeolian (minor), and Locrian modes refer to rotations of the same 7-note set, but with the root of the scale being at C, D, E, F, G, A, or B.

The standard notation for musical notes assigns A0 to the lowest note on the piano, C1 to the C above that, C4 to middle C, A4 to 440 Hz, etc., with octaves running from C to B. This is illustrated in Figure 4, which also shows the three new clefs proposed by Rossing.[Rossing] The highest clef is not for fundamentals. Rossing sticks with the convention of two different clefs. I cannot help but suspect that transposed treble clefs would be more convenient, but habit dies hard.

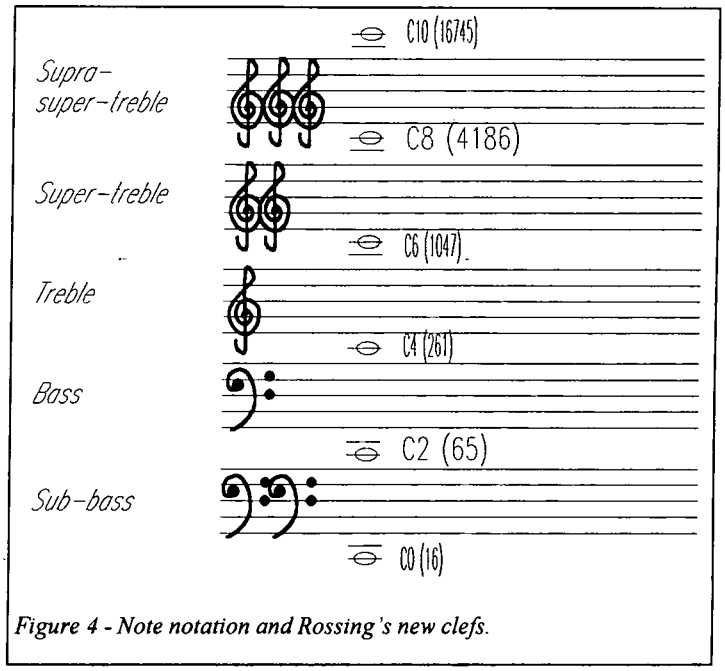


Figure 4 - Note notation and Rossing's new clefs.

2.1.4 The time domain

The time domain can also be resolved into two domains; linear and logarithmic. A sequence of notes will have onset times related to each other by approximately linear relationships, but the subdivisions of time are related logarithmically. For example, a typical song might have four phrases, each of which has four bars. Each bar (assuming 4/4 time) contains heavier stresses on minims, weaker stresses on crotchets, and further subdivisions into quavers and semiquavers. The periods here are related by factors of two. This is predominant in most types of music. Division into other primes is possible; 3 is common (much more so for divisions of crotchets into three triplet quavers than for divisions of phrases into three bars),

[9] Even for pieces in the 7-note diatonic scale, often the most commonly used notes are those in the pentatonic scale. Fleetwood Mac illustrate this particularly well.

[10] This excludes various scales termed Arabic, Jazz, whole-tone, etc.

but 5 and 7 are rare. Higher primes are most unusual for several reasons. First, they are hard rhythms for our perception to lock onto. Second, they tend to decompose into several unequal parts; a time-signature of 13/8 might be perceived as alternating bars of 6/8 and 7/8. In fact, composers who use such time-signatures usually specify this subdivision explicitly.

When there is a regular metre, the onset times are related to the bar times:-

$$(n+k)*T*\int(1/\text{tempo})$$

where n is an integer, T is the bar length, and k is a fraction. This fraction is usually nearly rational, and its denominator is a small, highly composite number such as 2, 3, 4, 6, 8, 9, 12, or 16, that stays fixed for long periods.

The durations of notes are generally notated in terms of the musical time – a legato note (on a driven instrument) may nominally be one crotchet in duration. However, the actual duration will be shorter for non-legato notes. Also, on non-driven instruments (percussion or plucked strings), where there is no control over the 'end' of the note, the duration is meaningless. The duration of a note is of less musical relevance than its start time.

The intention here is not to imply that all music has, or should have, the simpler distributions described above. Rather, we should say that, on observation, a surprisingly large amount of music tends to approximately follow these pseudo-rules for relatively long periods of time, weather permitting.

2.1.5 Redundancy

By identifying many floating-point numbers as being nearly integral, we have identified points where the data is compressible, at least in principle. To illustrate this, the decimal number 5.001234 is 101.0000000110101001 in binary. It is clear that all 'near-integers' have a format:-

(ones and zeros) . (all zeros or all ones) (ones and zeros)

This format is clearly compressible by using run-length encoding on the middle section. Thus, a stream of floating-point numbers that are likely to be near integers has, in theory at least, a lower information rate than an arbitrary stream of numbers. Likewise, the pseudo-types "small integer", "slowly-varying float", and "near-rational" have a format in which the data redundancy can be seen. It may not be practical to use this format, due to the complexity and cost of the coding and decoding.

2.1.6 Appreciation

Every 'rule' of music can be broken, and breaking the rules is how composers add interest and information to the music. However, when too many rules are broken at the same time, we reach a state of 'information overload', or 'musical chaos'. Here, our subjective appreciation of the music starts to fall, and we cannot use the preceding music to infer the future with much certainty. As the information increases, we ultimately reach noise, as shown in Figure 5.

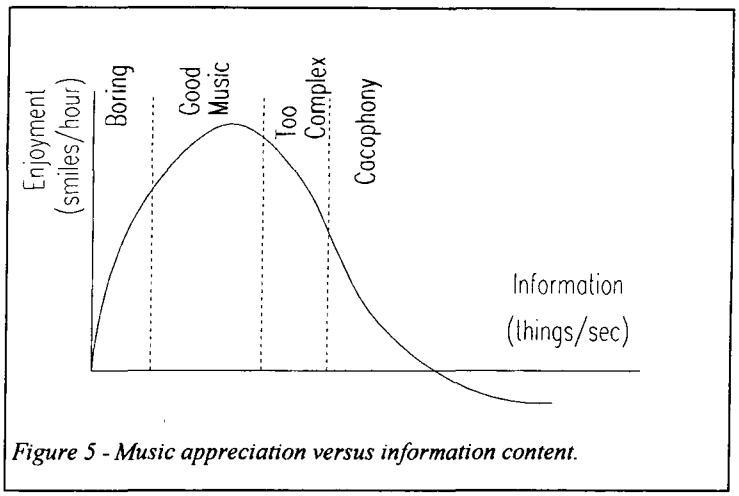


Figure 5 - Music appreciation versus information content.

The curve shown depends of course on the individual, the setting, the time of day, and a large number of other unknown parameters. However, the idea of information overload can be demonstrated easily by playing more than one piece of music simultaneously – this is instantly uncomfortable. The information rate obviously depends on tempo – excessively slow music tends to be boring, and excessively fast music (assuming no change in pitch) becomes uncomfortable.

2.1.7 Memory, context, and prediction

When listening to music, we interpret it with reference to both long-term and short-term memory. Our long-term memory is used to recognise an instrument, melody, orchestration, voice, and style as being familiar to something heard days or years ago. Short-term memory governs our appreciation of the current piece of music, and this applies to intervals of seconds (recognising a note as being the same pitch as the previous one) to minutes (interpreting a verse-chorus-verse-chorus structure) to hours (hearing a theme from the first movement repeated in the last). Here I use *context* to mean the interpretation of the current piece, as opposed to similarity to other pieces.

We are also able to predict what the music is likely to do in the future, and are pleasantly or unpleasantly surprised when these predictions do not come true. Repetitive instruments such as a bass drum beat blend into the background. We can also consciously choose to follow particular features in the music.

2.2 Perceptual properties of musical audio

2.2.1 Biology of the ear

Figure 6 - Georg von Békésy.

It is informative to know the physical structure of the ear. This has been studied for many years, most notably by von Békésy (1899-1972).[von Békésy]

The brief summary below borrows from Rossing and others.[Rossing, Neely, Wood, Plomp 76, MooreF 90, MooreB 95] The anatomy of the ear is roughly as shown in Figure 7. The *outer ear* passes external vibrations to the tympanic membrane, commonly referred to as the eardrum (also boosting the 2-7 kHz region). Rossing notes that the range of pressure variations is over six orders of magnitude, and that the displacement of the eardrum for the quietest note may be 10^{-8} mm = 10 picometres.

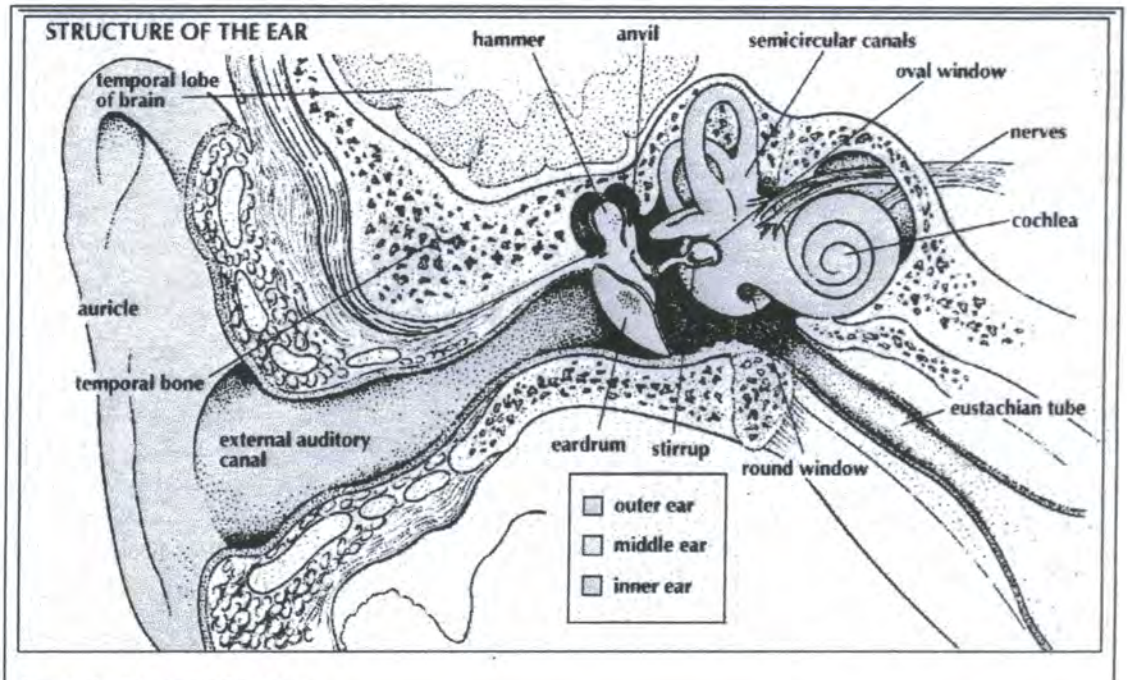


Figure 7 - Anatomy of the ear.

In the *middle ear*, the vibrations collected are passed via three minuscule bones called *ossicles*. These are known individually as the hammer, anvil, and stirrup (malleus, incus, and stapes). These act as a mechanical transformer; the lever action amplifies the force by about 1½, and the oval window is 20 times smaller than the eardrum. The pressure variations are thus amplified by a factor of 30. These bones also protect the inner ear from very loud sounds and sudden pressure changes. This process boosts

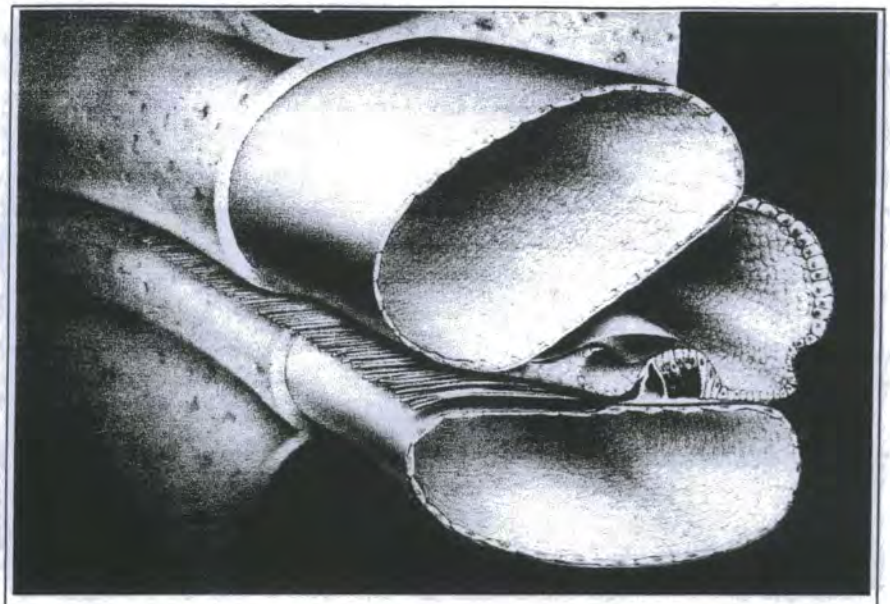


Figure 8 - Interior of the cochlea.

frequencies near 1 kHz. The middle ear also introduces various nonlinearities[Plomp 76] that give rise to aural harmonics and combination/difference tones.[11]

The *inner ear* contains the *semicircular canals* (our 3-d balance sensors) and the *cochlea*, shown in Figure 8. The cochlea can be thought of as a fluid-filled tube. Inside this tube are around 30000 hairs which vibrate at their own characteristic frequencies (CFs). The responses shown in Figure 9 show results of measurements on a cat's cochlea.[Kiang, Rossing] It will be noticed that the 'filters' are not symmetrical; they have a slower roll-off on the lower-frequency side (the 'tail') than on the higher-frequency side.[12]

[11] When frequencies f_1 and f_2 are presented to the ear, nonlinear transmission also causes terms f_1+f_2, f_1-f_2, $2f_1-f_2$, etc. to be perceived. These are known as combination or difference tones.[Pierce] It also causes $2f_1$, $3f_1$, $2f_2$, etc. – these are called aural harmonics.[Goldstein 67]

[12] Kiang comments on the dangers of applying data from anaesthetised cats to humans; however, the cochlear structures are similar, so one could reasonably expect similar results for humans.

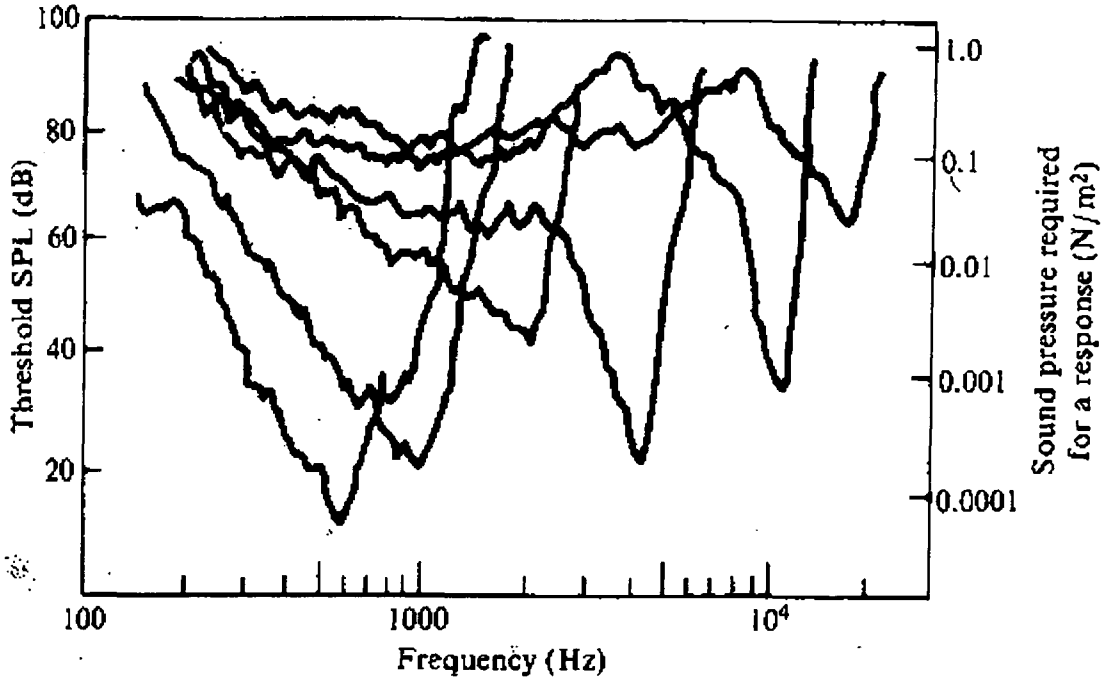


Figure 9 - Responses of six fibres in the auditory nerve of a cat.

The CFs are roughly evenly distributed by pitch, although gammatone filters are a closer approximation. The impulses are fed into the auditory cortex, and these are thought to then form 'maps' of amplitude and other parameters against frequency.[MooreD, BrownG 92a, BrownG 94a]

2.2.2 Critical bands and masking

When two tones overlap in frequency such that their responses overlap considerably, they are said to lie in the same *critical band*.[Rossing, Zwicker] The *critical bandwidth* is around a third of an octave for much of the frequency range, but is greater at low frequencies. An approximate linear expression for the critical bandwidth at f Hz is (f/9+80) Hz, though the data does not fit a line perfectly.

2.2.3 What are the musical atoms?

We still have the decidedly non-trivial problem of defining the basic entities of music. How many partials must a note have? How much frequency stretching is permissible? Is a timpani roll one note or many? Is a violin section a single instrument or sixteen? When is the onset of a reversed cymbal crash? How can we recognise notes as wrong or missing without any prior knowledge of the score? When does a repeated motif become part of the background? Is Haydn's 'Surprise Symphony' a different piece of music the second time round? What is the sound of one hand clapping, and how much disk space does it require? A computer model can at least attempt to answer some of these questions in an objective and repeatable manner, and some limitations of the model developed later are analogous to limitations of our own perception. Whether we can use the reactions of an artificial listener to infer anything about our own listening is of course debatable.

2.3 Source separation and stereo imaging

The aspect that makes polyphonic transcription so much harder than monophonic transcription is that we effortlessly separate the two channels into their component notes. The sound reaching our ears includes noises that we identify as being entirely unrelated to the music, such as the noise of traffic or the hum of electrical devices. It has been thought that we dissociate separate sources by lower-level processes. However, some studies[BrownG 94a] have pointed out that this mechanism could also form the basis of source separation and timbre identification.

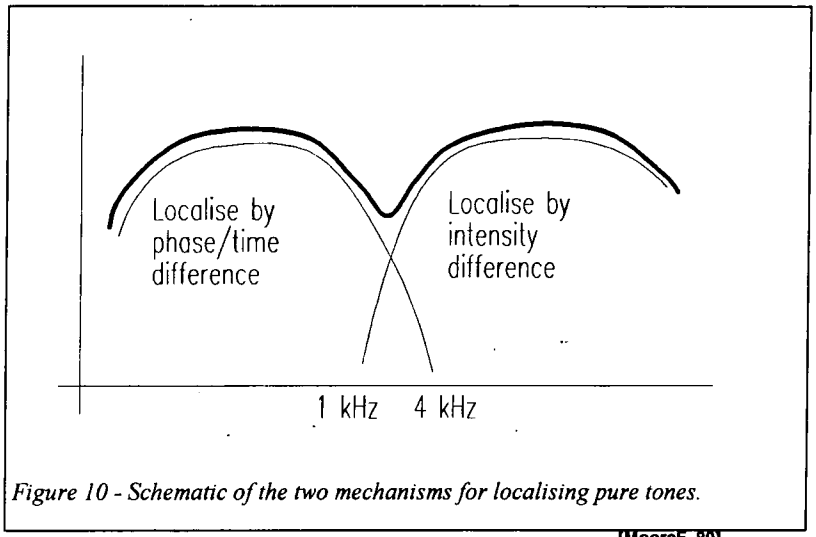


Figure 10 - Schematic of the two mechanisms for localising pure tones.

The preliminary processes also involve identification of the position of the source. We judge this by the interaural phase difference and the interaural time difference (ITD) for tones below 1 kHz and the relative amplitudes at each ear (the interaural intensity difference or IID) for tones above 4 kHz.[MooreF 90] On the left-right axis, our accuracy is greater for high notes than for low notes, and has a dip at around 3 kHz. This is illustrated in Figure 10. Kendall gives a more detailed overview of directional hearing.[Kendall 91, Kendall 95] The complex shape of our outer ear helps to distinguish up/down and back/front, although in these axes our accuracy of placement is much poorer.[MartinK 95]

Another aspect of real music to bear in mind is that there are usually effects such as reverberation and filtering applied to the signal. Our preliminary auditory processes are largely responsible for determining the distance and environment of a sound. With acoustic reverb, the impulse response of the room is constant but very complex. In a digital environment, even more effects can be applied to the sound, and these can create sonic environments that are physically unrealisable, such as a sound that reaches the left ear first but is stronger in the right ear. Frequently, different effects are applied to different instruments.

2.4 Perceptual properties of notes

Most music is created and perceived in terms of individual notes. As usual, there are exceptions – music can be organised as slowly changing textures, without any concept of onsets and offsets. Individual notes can be studied in isolation more readily than polyphonic music. The physical parameters of audio waveforms have analogies in the perceptual parameters, but the relationships between them are often very complex and interdependent. The table below, after Rossing, shows the dependence of perceptual

parameters on physical parameters – note that every perceptual parameter depends on every physical parameter.[Rossing]

	Loudness	*Pitch*	*Timbre*	*Duration*
Pressure	**STRONG**	weak	weak	weak
Frequency	weak	**STRONG**	moderate	weak
Spectrum	weak	weak	**STRONG**	weak
Duration	weak	weak	weak	**STRONG**
Envelope	weak	weak	moderate	weak

Table 2 - Dependence of perceptual parameters on physical parameters.

In addition, subjective onset time may vary from physical onset time.[Wessel 78]

2.4.1 Amplitude – loudness

Listeners can reliably and repeatably adjust sounds to 'twice as loud', 'half as loud', and there is little variation between people.[Sundberg 91] However, what is it that people are describing by loud? The relations between loudness and physical factors are complex, and there are several competing definitions. Below is a brief summary.[Rossing]

The *sound pressure level* (SPL) is defined from the logarithm of the ratio of a sound's pressure to a reference pressure of 20 μPa. All logarithms are to base 10.

$$L_p = 20 \log (p/p_0)$$

This gives the following levels:-

Source	*SPL (dB)*	*quality*	*pressure*
Jet takeoff at 60 m	120		20 Pa
Construction site	110	intolerable	6.3 Pa
Shout at 1½ m	100		2 Pa
Heavy truck at 15 m	90	very noisy	630 mPa
Urban street	80		200 mPa
Car interior	70	noisy	63 mPa
Normal conversation at 1 m	60		20 mPa
Office/classroom	50	moderate	6.3 mPa
Living room	40		2 mPa
Bedroom at night	30	quiet	630 μPa
Broadcast studio	20		200 μPa
Rustling leaves	10	barely audible	63 μPa
Silence	0		20 μPa

Table 3 - Sound pressure levels of several sounds.

The *sound power level* is the total power emitted by a source in all directions. It is defined by:-

$$L_W = 10 \log (W/W_0)$$

where W_0 is one picowatt.

The following table from Wood gives the powers of several instruments.[Wood] These are maximum values unless otherwise stated.

Source	*Power*	*SPL (dB)*
Orchestra of 75	70 W	138.5
Bass drum	25 W	134.0
Pipe organ	13 W	131.1
Snare drum	12 W	130.8
Cymbals	10 W	130.0
Trombone	6 W	127.8
Piano	400 mW	116.0
Bass saxophone	300 mW	114.8
Bass tuba	200 mW	113.0
Double bass	160 mW	112.0
Orchestra at average loudness	90 mW	109.5
Piccolo	80 mW	109.0
Flute	60 mW	107.8
Clarinet	50 mW	107.0
French horn	50 mW	104.8
Triangle	50 mW	104.8
Bass voice	30 mW	104.8
Alto voice ***pp***	1 mW	90.0
Average speech	24 μW	73.8
Violin ***ppp***	3.8 μW	65.8

Table 4 - Sound power levels of several sounds.

The *sound intensity* is the rate of energy flow across a unit area, relative to 10^{-12} Wm^{-2}.

The *loudness level* L_L of a sinusoid is the sound pressure level of an equally loud 1000-Hz sine. This is strongly dependent on frequency, as shown by the Fletcher-Munson curves of equal loudness in Figure 11.

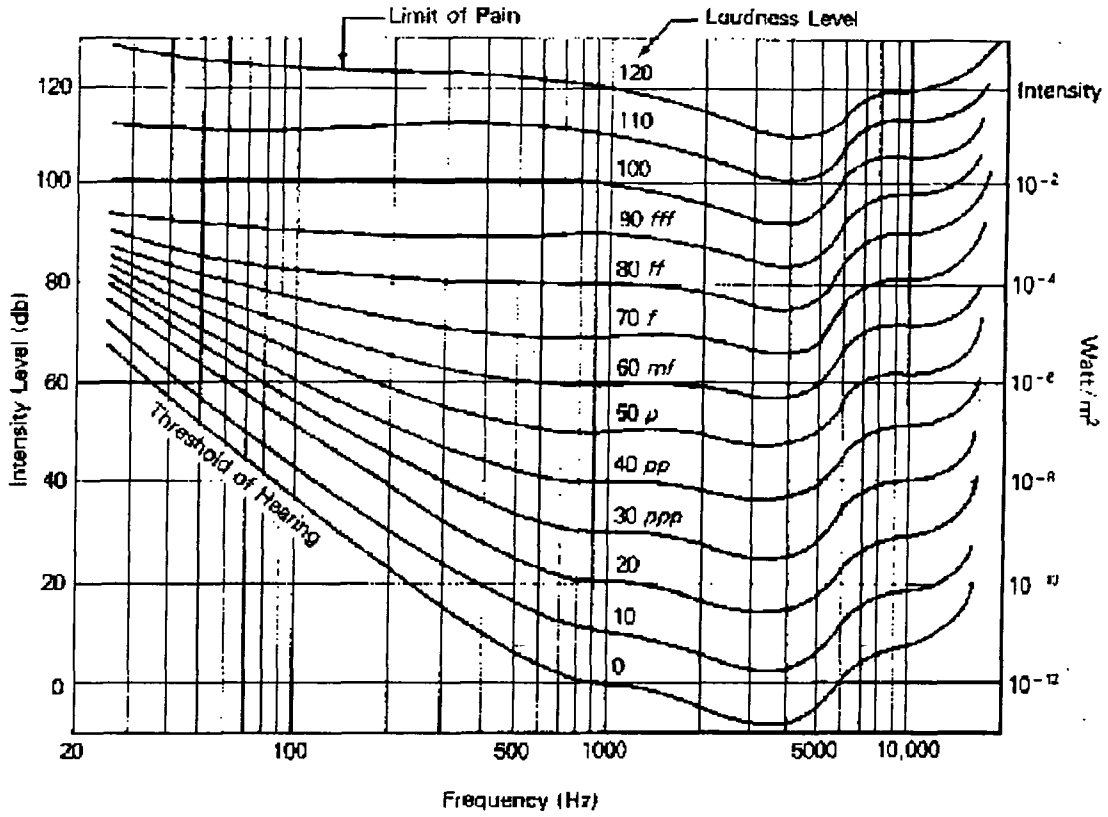


Figure 11 - Fletcher-Munson curves.

The loudness of a complex tone depends on the loudnesses of its components, and whether they fall into the same critical band.

2.4.1.1 Loudness vs. spectral cues

We have discussed the main factor in loudness, intensity. John Chowning discusses how loudness also depends on other factors.[Chowning 93] He illustrates with the example of two singers; one singing *forte* at a distance of 50 metres, and another singing *pianissimo* at 1 metre. The listener will judge the distant sound to be louder, despite the fact that the close sound has an intensity 2½ times greater. To do this, we must first judge distance, and von Békésy shows that this is primarily dependent on the ratio of direct sound to indirect sound.[von Békésy]

2.4.1.2 Dynamic range

The dynamic range of an instrument is defined as the difference in loudness between the loudest and the quietest notes. Several values are shown in the table.[PattersonB]

Instrument	*Dynamic range (dB)*
Recorder	10
Double bass	30
Flute	30
Trombone	38
Bassoon	40
Violin	40
Clarinet	45

Table 5 - Dynamic ranges of several instruments.

2.4.1.3 Amplitude resolution

Rossing estimates that our amplitude resolution is 1.7 dB.[Rossing] MIDI keyboards have 127 discrete velocity levels, which would imply around 1-dB steps, and no complaints are raised about the lack of resolution. A lower figure still is suggested by Rasch, though, who states that "SPL differences of less than 1 dB ... can have a profound effect on the subjective response to a stimulus".[Rasch 82] Sundberg gives the lowest estimate of 0.43 dB.[Sundberg 91] However, Risset comments that the room response fluctuations of up to 20 dB are much greater than our amplitude resolution.[Risset 82]

2.4.2 Frequency – pitch

Pitch is defined[ANSI] as *"that attribute of auditory sensation in terms of which sounds may be ordered on a scale extending from low to high"*. While few would disagree with this, it is not obvious why we assign the terms "high" and "low" to pitches in the first instance.[13] Although pitch is nominally treated as the logarithm of frequency, it is a more complex phenomenon. Most people do not have absolute pitch[Ward] (otherwise known as perfect pitch), although some learn it. For many, the lowest pitch they can sing is a fairly stable reference, and others imagine certain notes for instruments or pieces in distinctive keys.[14] Most can only recognise relative pitch – i.e. the ratio of two frequencies. Sundberg draws attention to the concept of *extra-musical pitch*, such the pitch of voices, sibilance, drums, and cymbals.[Sundberg 91] Plomp uses the term *nontonal pitch*.[Plomp 76] Plomp also points out the phenomenon of *binaural diplacusis*, whereby the pitch in one ear may differ from that in the other, by up to 2% – a third of a semitone.

The prevailing standards define A (A4) above middle C (C4) to be 440 Hz.[Wood] The 'old' notation for C1/C2/C3/C4/C5 is C_1 /C/c/c'/c''. The *cent* is a convenient interval of a hundredth of a semitone, equal to $\sqrt[1200]{2}$.

2.4.2.1 Psychophysical pitch

There are two scales on which psychophysical pitch can be measured. The unit of subjective pitch is the *mel*, where doubling the number of mels "doubles the tone height". A different scale defines one *bark* as equal to the critical bandwidth, and this gives one bark as very close to 100 mels. However, the mel and bark scales are of little use as they distort the low end of the spectrum compared to the linearity of the log-frequency definition.

---

[13] In this context, an interesting story is told of a 6-year-old girl thumping on the bass end of a piano; when an adult suggested that she played at the high end, she replied that she was already at the high end because the notes 'sounded so much bigger' than those produced by the keys to her right. This agrees with Dowling's comments on young children's pitch perception.[Dowling]

[14] For this author, common 'tuning references' include the C# key of the "Moonlight Sonata", the Eb tonic of the Mozart horn concertos on an Eb tenor horn, and the EADGBE of a guitar.

2.4.2.2 Pitch resolution

Pitch resolution depends on several factors such as frequency, sound level, duration, smoothness of transition, measurement method, and musical ability.

For pure tones, it has been shown[Zwicker, Rossing] that the just-noticeable difference (JND) between two frequencies is about a twelfth of a semitone from 1000 to 4000 Hz (although at 2000 Hz there is a slight dip). From 1 kHz to 4 kHz, we can resolve pitch to around 0.05%, or a twelfth of a semitone, or one thirtieth of the critical bandwidth (the JND depends on the frequency, duration, sound level, and the abruptness of the change). Rossing comments that we can thus hear around 5000 *different* frequencies, but Sundberg gives a lower figure of 1400.[Sundberg 91]

The resolution is poorer at lower frequencies, and is never better than 2 Hz. Throughout most of the range, there are about 30 JNDs in one critical band. The table below summarises the results.[Zwicker]

frequency (Hz)	*JND (Hz)*	*JND (cents)*	*critical band (Hz)*	*JNDs per critical band*	*semitone (Hz)*	*JNDs per semitone*
60	3	84	90	30	3.57	1.19
100	2.4	40	90	37.5	5.95	2.48
200	2.5	21	90	36	11.90	4.76
500	3	10.1	110	36.7	29.74	9.91
1000	4	6.7	150	37.5	59.5	14.88
2000	10	8.4	280	28	119.0	11.90
5000	20	6.7	700	35	297.4	14.87
10000	80	13.4	1200	15	595	7.44

Table 6 - Critical bandwidths and JNDs at various frequencies.

Rakowski shows that under ideal conditions, pitch changes of 0.03 to 0.08 Hz can be noticed at 160 Hz. This corresponds to a much better resolution of one three-hundredth of a semitone.[Rokowski, Jansen 91] It is unclear, however, what these conditions are.

2.4.2.3 Pitch and amplitude – Stevens's Rule

In 1935, Stevens showed that low tones get flatter as they get louder and high tones get sharper.[Stevens] This is sometimes termed *Stevens's Rule*. The maximum shifts were recorded at 150 and 8000 Hz. The transition, where pitch is independent of loudness, is at 2000 Hz. Measurements[Terhardt 79] at frequencies between 200 and 6000 Hz show that the amount of shifting is approximately linear, and agrees closely with the following approximation, shown in Figure 12, where f is the frequency, L_P is the sound pressure level in dB, $pitch_{60dB}$ is the pitch at L_P=60 dB, and 'Stevens's constant', $K_{st} \approx 3.362 * 10^{-4}$ cents per decibel per Hertz.

$$pitch = pitch_{60dB} + K_{st}*(f-2000)*(L_P-60)$$

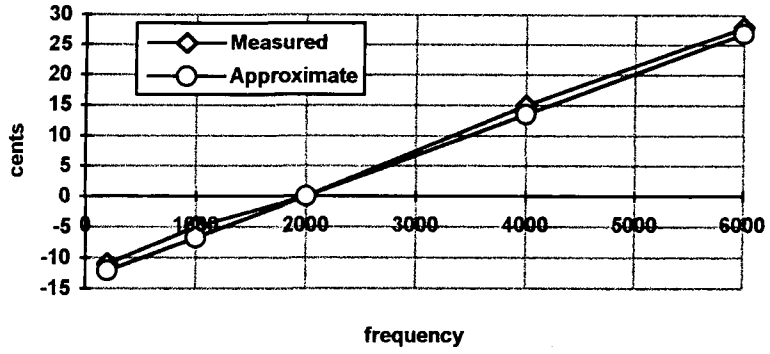


Figure 12 - Pitch variation in cents per dB amplitude change.

For a 200-Hz *pure* tone, an increase in amplitude of 40 dB (comparable although not equivalent to the difference between *pp* and *ff*) flattens the pitch by 22 cents, nearly a quarter of a semitone. A 6-kHz tone would be sharpened by 56 cents, and would be closer to the semitone above.

Although this effect is noticeable for sine tones, the pitch of complex tones depends little on intensity. Complex tones will have harmonics both below and above 2 kHz.

2.4.2.4 Pitch and duration

In 1840 Savart first examined how long a tone must last for in order for it to be a pitch rather than a click. His results suggested that 2 cycles were sufficient, but later experiments gave higher figures, and showed that the number of periods increases with increasing frequency. From 50 to 1000 Hz, the number increases slowly from 3 to 11, corresponding to the recognition period falling from 60 to 11 ms. It then increases rapidly to 250 at 10 kHz. Truax quotes Olson's experiments showing that pitch develops after 13 ms.[Olson, Truax 88]

2.4.2.5 Pitch of complex tones

The pitch of a complex tone is determined by the pitches of each harmonic.[Goldstein 73, Terhardt 79, Terhardt 82b] When the fundamental is absent, the sensation of pitch is still clear (for fundamentals under 1000-2000 Hz[GreenD]) – this is an example of *virtual pitch*, and such a tone is known as a *residue tone*. Rossing cites experiments showing that the fourth and fifth harmonics are the most important for determining pitches up to 200 Hz. For higher pitches, lower harmonics are more important, and above 2500 Hz, the fundamental is the most important. Rasch describes this as a 'dominance region' for partials to affect pitch, and also quotes Terhardt's results showing that a complex tone is heard as being lower than a pure sine at the same frequency.[Rasch 82, Terhardt 71]

2.4.2.6 Frequency stretching

The frequency of a complex tone with not-quite-harmonic partials is harder to specify, but the brain is assumed to find a set of "nearly harmonic" partials and decide a fundamental.[MooreB 86, Piszczalski 79] On many instruments, including strings[Shankland, Fletcher 62, Fletcher 77a, Fletcher 77b, Blackham, Schuck, Rossing, Kottick,

[Kurz], organ pipes[Fletcher 63], and flutes[Benade 66], the partials are approximately, but not exactly, harmonic. The most common example of this is in low strings on an upright piano, which are very thick and insufficiently long.[15] As a result the relative frequency rises faster than the partial number.[16] Stiff strings' harmonics are given by the following equation:-

$$\frac{f_n}{f_1} = n\left[1 + \frac{\pi^3 r^4 Y}{8l^2 T}\left\{\left(n^2 - 1\right) + \frac{C}{r^2}\left(A_1{}^2 - n^2 A_n{}^2\right)\right\}\right]$$

where f_n is the frequency of the n^{th} partial, r is the string radius, Y is the Young's modulus, l is the length, T is the tension, C is a 'bridge parameter' of around 0.27, and A_n is the amplitude of the n^{th} harmonic. Rossing gives the simpler version of this formula[Rossing], with C=0:-

$$\frac{f_n}{f_1} = n\left[1 + \frac{\pi^3 r^4 Y}{8l^2 T}\left\{n^2 - 1\right\}\right]$$

Data from tests by Blackham on the lowest note, A0 (27.5 Hz), is shown in Figure 13, and data from tests by Schuck on a higher note, F1 (44 Hz), on a different piano, is shown in Figure 14.[Blackham, Schuck]

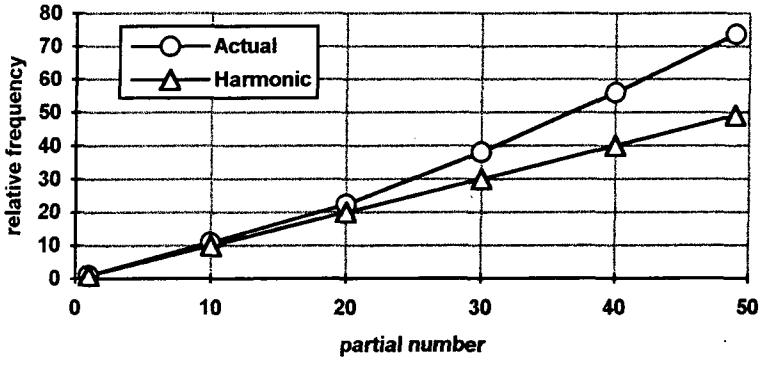


Figure 13 - Inharmonicity of A0 from Blackham's data.

[15] Railsback examines how this affects actual piano tuning.[Railsback, MartinD] However, inharmonicity does not appear to affect organ tuning.[Sundberg 73]

[16] I am not aware of any acoustic instruments exhibiting the opposite phenomena of 'frequency squashing' – where higher partials become *flatter*.

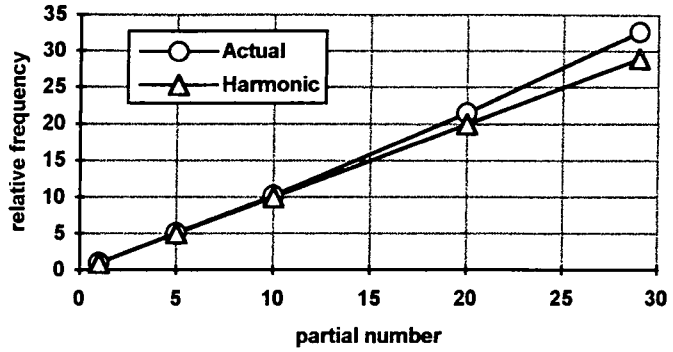


Figure 14 - Inharmonicity of F1 from Schuck's data.

The above has discussed *frequency* stretching, but another factor counteracts this. Given one tone and asked to tune another to an octave above, listeners will actually stretch the octave to an average of around 1215 cents, nearly a sixth of a semitone sharp. This sharpness depends on frequency, amplitude (due to Stevens's rule), and spectrum.[Sundberg 91, Sundberg 73] Our log for *perceptual pitch*, as opposed to frequency, should be 2.0174 rather than 2. Perfectly harmonic tones would actually be slightly pitch-squashed.

2.4.2.7 Inharmonicity

The pitch of timpani, and to a lesser extent bells, are borderline cases between pitched and unpitched. The pitch is implied by approximate relationships between some of the partials, but the sensation of pitch is less clear. In some cases, there may be a sensation of several pitches, as in some bells, or no pitch, as in a bass drum.

2.4.2.8 Discussion of contrabass instruments

Acoustic contrabass instruments are difficult to construct due to their size, and low notes are often difficult to produce. In many cases there is little energy at the fundamental, which is instead implied by the higher harmonics, which have been shown to be more relevant to pitch than the fundamental. As an alternative to using a single low note of, say, 30 Hz, it is common for composers to use fifths – 60 and 90 Hz – to "thicken the bass" by implying a pitch an octave below.

The lowest notes on several contrabass instruments are listed in the table below. We can assume that most music goes down to 30 Hz and in extreme cases to 15 Hz.

Instrument	*Lowest note*	*Frequency (Hz)*	*Period (ms)*
Human voice	A 1	55	18.1
Double bass and contrabass violin <1>	E 1	41.2	24.3
Bass guitar	E 1	41.2	24.3
Harp	C 1	32.8	30.8
Subcontrabass recorder	F 2	87.3	11.5
Double contrabass flute <2>	C 1	32.8	30.8
Contrabass clarinet (BBb)	D 1	36.7	27.2
Octocontrabass clarinet (BBBb) <3>	D 0 (C 0)	18.3 (16.4)	54.6 (61.6)
Bass saxophone (Bb)	Ab 1	51.9	19.2
Contrabass saxophone (Eb) <3>	Db 1	34.6	28.9
Bassoon	Bb 1	58.2	20.4
Contrabassoon <4>	Bb 0	29.1	44.9
Eb Bass tuba	A 0 (E 0) <5>	27.5 (20.6)	36.4 (48.5)
BBb (Contrabass) tuba	E 0 (B -1) <5>	20.6 (15.5)	48.5 (64.5)
Bass marimba, bass steel drum	C 2	65.6	15.4
Piano	A 0	27.5	36.2
Organ	A -1 (C -1) <6>	13.8 (8.12)	72.4 (123.2)

Table 7 - Frequencies of contrabass instruments.

<1> The contrabass violin is the largest of the new viol family developed by the Catgut Acoustical Society.[Rossing]

<2> This modern instrument is three octaves below the standard flute.

<3> Contrabass saxophones are extremely rare – Grant Green, a contrabass woodwind fanatic, reports that only fifteen are known to exist. There are three octocontralto (EEEb) clarinets and only one octocontrabass (BBBb) clarinet.[GreenG]

<4> The record for a double-reed instrument is a 'sub-double' bassoon, no longer surviving.[Matthews] This is an octave below the contrabassoon, and reaches down to B-1 (14.6 Hz). The EEb contrabass sarrusophone, still used in continental Europe, reaches Db1 (34.6 Hz).

<5> A0 is the lowest pedal note (first mode) on a 3-valve Eb tuba. This is the lowest note in the symphonic repertoire[Bevan], and can be considered the bottom of the tuba range. A compensated 4-valve Eb tuba can *theoretically* play E0, but the low end of the pedal-note register is very difficult to play. The BBb tuba (the nomenclature 'contrabass tuba' is not often used) would reach E0 (3-valve) or B-1 (4-valve). The rare BBb contrabass trombone can also reach E0. There are a few EEb ('subbass') and two BBBb subcontrabass tubas, the last theoretically capable of E-1 (10.3 Hz).[Baines] The obsolete contrabass ophicleide reached Eb1 (38.9 Hz).

<6> The C-1 figure is exceptional[Matthews], from a 64-ft pipe, and would not be found on most organs. The lowest organ notes are often made using two shorter pipes with fundamentals corresponding to the 2nd and 3rd harmonics, a practice developed by Abbot Vogler (1749-1814).[Wood]

The difficulties of constructing and playing physically large instruments do not apply to electroacoustic synthesis as computer models do not have such physical restrictions. (Low notes may still be problematic – for example, a sampler needs more memory, and a recursive oscillator is more susceptible to round-off error due to the small phase increment.) Composers have been able to explore the transition region between pitch (such as a snare drum roll played double speed) and rhythm (the same roll played

half speed). The notion of a 'lowest possible frequency' is thus impossible to define. Frequency does not always have to be seen as high or low; sometimes it has its absolutely literal meaning of 'very often' or 'not very often'. This will be familiar to those working with wavetable synthesis and pitch-synchronous granular synthesis.

Figure 15 - Hermann von Helmholtz.

2.4.3 Spectrum – timbre

Helmholtz (1821-1894) was the first to assert that the timbre of the steady-state portion of a sound depends on its spectral characteristics.[Helmholtz] Relatively pure tones, such as flutes, and tubas in the high register, sound dull. When most of the energy is in lower harmonics, the sound may be described as dull or mellow or bassy. When higher harmonics have more energy, the sound is bright. Some instruments, notably clarinets, have stronger odd harmonics, and this causes 'reediness'. The steady-state timbres of many orchestral instruments have been analysed and made available by Gregory Sandell.[Sandell 91, Sandell 95] However, Fourier's explanation is really only true during the steady-state portion of the sound. Indeed, there may be no 'steady state', as the temporal variation of these parameters contributes greatly to the qualities of a sound. It is also known that partials will only fuse if their rates of vibrato are the same.[Dubnov]

Several studies have used multidimensional scaling to determine what the most important aspects of timbre are.[Plomp 76, Grey 77a, Wessel 78, Wessel 79, Rossing, De Poli 93, Toiviainen] Miller showed that subjective similarity depends on the envelope.[Miller] Plomp gives the following 'dissimilarity matrix' for nine instruments[Plomp 70], and maps the differences onto three dimensions. Darker shading represents higher similarity between instruments.

	violin	viola	cello	oboe	clarinet	bassoon	trumpet	horn	trombone
violin	0	89	25	100	19	90	88	101	96
viola		0	87	34	115	31	46	62	79
cello			0	92	40	40	98	72	38
oboe				0	107	76	74	26	67
clarinet					0	78	124	97	78
bassoon						0	87	36	6
trumpet							0	103	109
horn								0	10
trombone									0

Table 8 - Dissimilarity matrix for notes.

Von Bismarck's study[von Bismarck] gives dull/bright, cold/warm, pure/rich, and Plomp's[Plomp 76] gives dull/sharp, compact/scattered, full/empty, colourful/colourless. Dull/bright has appeared as the main factor in every study of timbre. Terhardt contributes 'roughness', corresponding to spectral line-

widening.[Terhardt 78, Risset 82] Ethington lists 124 adjectives applied to timbres and presents a system that implements 'text-to-timbre' for 16 of them, namely blown, bowed, hammered, keyed, plucked, struck, damped, legato, percussive, staccato, sustained, resonant, clear, ringing, bright, and surging.[Ethington] This allows the user to change from one timbre to a "slightly more plucked, much less resonant, more damped" timbre.

Risset cautions against placing too much reliance on the spectrum as a means of identification, pointing out that the response of a normal room has fluctuations of up to 20 dB.[Risset 82] This varies with position, but our moving around in a room does little to alter our perception of timbre.

Timbre was also shown[Miller, Grey 77a, Wessel 78] to be strongly connected to the order and rate with which the harmonics start. On a plucked string, for example, the higher harmonics start before the lower harmonics. On brass instruments, the reverse is true.[Risset 69]

Diana Deutsch examined grouping mechanisms.[Deutsch] She showed that timbre is very dependent on the attack of the sound, by splicing the attack of a trumpet (~20 ms) onto the sustain part of a clarinet, and showing that the timbre was perceived as a trumpet. Berger carried out similar experiments, removing the first and last half seconds of a note on 10 instruments and asking listeners to identify them.[BergerK] The result was the following 'confusion matrix'. Darker shading represents more identifications.

	Response										
Stimulus	Flute	Oboe	Clarinet	Tenor sax	Alto sax	Trumpet	Cornet	French horn	Baritone	Trom-bone	No answer
Flute	1	2		1	6	5	4			4	7
Oboe		28									2
Clarinet	1	1	20	4	3						1
Tenor sax			25	2	1						2
Alto sax				3	4		1	11	5	5	1
Trumpet	8				6	2	3	4	1	3	3
Cornet		1				12	15				2
French horn	1			2	3			5	6	6	7
Baritone			1	1	2	3	2	4	7	3	7
Trombone	2	1		5	3			1	5	9	5

Table 9 - Confusion matrix for notes without onsets and offsets.

Alfred Bregman also examined auditory grouping in detail.[Bregman 89] He identifies the characteristics linking auditory elements as common location, harmonicity, common onsets and offsets, and common frequency and amplitude modulation.

2.4.3.1 Timbre resolution

Timbre being a multidimensional and time-varying quantity, it is hard to quantify how much resolution of timbre space we have. It is clear that only a very short duration is needed. Robinson showed that

listeners can distinguish between brass, flute, harpsichord, and strings given *a single period of the note*[RobinsonK], and Gray showed above-chance performance for identifying a vowel from a single period.[Gray]

2.4.3.2 The sound of an instrument

An important question in both analysis and synthesis is "What is the sound of a trombone?", or any known instrument. For analysis, we wish to know "What does this sound data represent or mean in terms of higher-level parameters such as pitch, loudness, duration, bite, meatiness, roundness, etc.?". For synthesis, we wish to know "What function of pitch, loudness, duration, bite, meatiness, roundness, etc. would create a sound similar or identical to what a trombonist and a trombone would create?".

The spectrum of a note on a particular instrument depends on its frequency – often high notes have most of their energy in the first few harmonics, but low notes have very little. In many cases, the prominent range of frequencies is nearly fixed. This is known as a formant, especially in reference to the human voice. In general, our perception of timbre depends more on formant structure than a theory based on overtones. A low 'ah' and a high 'ah' have similar timbres because they have roughly[17] the same formants, even though the relative strengths of overtones are different.[Slawson, Plomp 76, Risset 82, Cook 91] The frequencies of speech formants[MooreF 90] are illustrated in Figure 16.

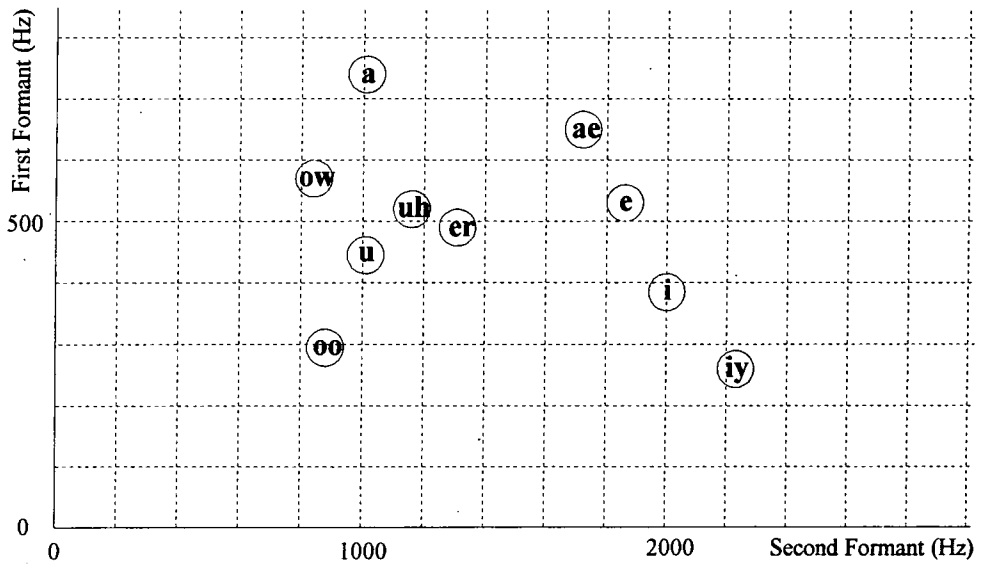


Figure 16 - Vowel formants.

Many, indeed most, other instruments also have formants. It has been suggested, in the context of a Stradivarius violin, that a third formant at the sum of the frequencies of the first two is an important

[17] Slawson suggests that if the fundamental frequency doubles, the formants should rise by 10%.

parameter in subjective timbre quality.[Dubnov] This can be achieved by non-linear effects devices which, amongst other spectral alterations, have the effect of adding a 'third formant' to an instrument with two.

The spectrum also depends on the precise manner of playing a certain pitch. A (sounding) F4 played in sixth position on the eighth mode of a trombone will have a brighter spectrum than one played in the first position on the sixth mode. A trumpet with all valves down has different acoustics to one played open, both because the cylindrical section is longer and because the bends in the tube add fluid impedance (and as discussed earlier, it also has poor intonation). Different ways of fingering also affect keyed woodwind[Sandell 91] and keyed brass[18] in similar ways. Likewise, the top E4 string of a guitar has a mellower and more harmonic timbre than one played on the 24th fret of the low E2 string. Both research and compositional experience show that modelling the playing structure of an instrument with special attention to the valves[19], slides[20], keys, or frets adds a great deal of realism.

The spectrum also depends on amplitude – louder notes generally have more energy in higher harmonics. Even if these can be deduced, the spectrum, and the envelope of each harmonic, depends on the actual articulation used for each note. Some instruments can make distinct sets of timbres. A good example is the clarinet, which has three registers (chalumeau, clarion, and altissimo) that have different timbres. However, in listening to music, we invariably can tell that one group of notes came from one instrument, whether a familiar or a new instrument.

2.4.4 Phase

In almost all cases, we cannot hear a difference when the relative phases of the harmonic are changed.[Risset 69] This insensitivity to phase is termed 'Ohm's acoustical law'. A few experiments have found counter-examples.[Licklider, Plomp 76, Van Klitzing, Leman 94] Plomp concludes that the maximum effect is between a tone with harmonics in phase (sine or cosine summation) and a tone with harmonics differing by 90°. Wang proves that arbitrarily many zero-crossings may be introduced into a wave with the same magnitude spectrum as a square wave.[Wang]

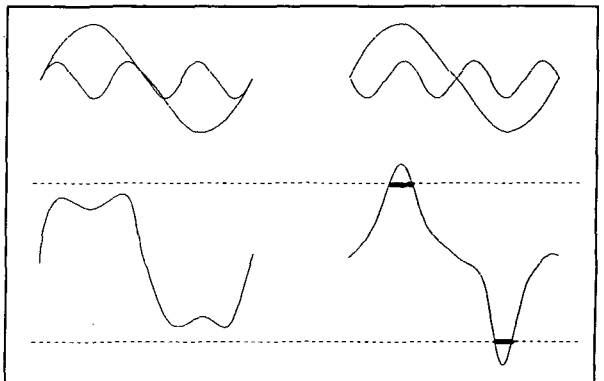


Figure 17 - Non-linear distortion.

Another possibility is that some non-linear distortion is taking place in the ear. This is

---

[18] Keyed brass are now rare; they include ophicleides and bugles.

[19] The Yamaha SY77 allows each note to be tuned individually. This allows the faulty intonation of an uncompensated brass instrument to be modelled more realistically.

[20] Processing has been implemented (using Cakewalk's CAL language) to map a MIDI trombone line so as to add portamento corresponding to the slide position for each note. Again, this subtle effect adds realism.

schematically illustrated in Figure 17. In this example, the tone has only the first and third harmonics, but the absolute value of the output is clipped. Whether the signal is clipped depends on the relative phases of the harmonics. It is known that the middle ear does introduce some non-linearities, but it is unlikely to be as simple as clipping.

2.4.5 Time – musical time

The timings of notes are interpreted using an overall tempo and a hierarchical metrical structure, as discussed earlier.

2.4.5.1 Time resolution

The order of onsets can be judged with an accuracy of around 1-2 ms[GreenD], although this accuracy decreases as the frequency falls. Scheirer quotes Handel's figure of 5 ms.[Scheirer 95b, Handel]

Rasch showed that onset asynchrony is an important cue in the perception of music.[Rasch 78] He carried out threshold experiments on distinguishing whether a motif is played up or down in the presence of a masking tone. The threshold fell by 35 dB when the asynchrony was 20 ms. However, up to 30 ms, subjects did not report any perceived asynchrony. This agrees with data from research by Hirsh and comments by Moore.[Hirsh, MooreF 88] Informal tests showed that the accuracy of actually playing a musical example on a keyboard varies from 10-30 ms, and the accuracy of playing repeated notes is around 5-15 ms.

2.4.5.2 Heisenberg's uncertainty principle

The uncertainty in time and the uncertainty in frequency are related by Heisenberg's uncertainty principle. Where Δt_s and Δf_s are the 'inertial' width, $\Delta t_s \times \Delta f_s \geq 1/(4\pi)$. This is derived in Appendix G.[Solbach 96b, Papoulis] Gabor chooses a different definition, where Δt and Δf are $2\sqrt{\pi}$ times Δt_s and Δf_s respectively.[Gabor 47] With these definitions, $\Delta t \times \Delta f \geq 1$. But how closely does our auditory system come to this inequality?

Gabor quotes data from perceptual tests by Bürck giving $\Delta t \times \Delta f = 2.1$ at 500 Hz and 3.0 at 1 kHz[Bürck], and data from Shower giving $\Delta t \times \Delta f = 2.34$.[Shower] Srinivasan's figures give $\Delta t \times \Delta f = 2.07$.[Srinivasan] Later data from Majerník suggests that $\Delta t \times \Delta f$ can be less than 1.26.[Majerník] It seems that our perception is nearly as sharp as possible. Gabor notes, "the best ears in the optimum frequency range can just about discriminate one acoustical quantum". Mont-Reynaud suggests, without stating sources, that the ear can even beat the uncertainty principle.[Mont-Reynaud 93]

For complex tones, the situation is different. Winckel showed that a sense of pitch can develop for a note as short as 3 ms[Winckel], although Bürck's data shows this interval falling from 60 ms (3 periods) at 50 Hz to 18 ms at 400 Hz (7.2 periods) to 13 ms at 2000 Hz (26 periods) then rising to 28 ms at 10000 Hz (280 periods).[Bürck] It is likely they used differing definitions and/or experimental setups. Robinson

showed that 4-32 cycles are needed to reliably determine the pitch chroma (sufficiently to distinguish C/D/E/F), which was longer than it took to identify the instrument.[RobinsonK]

2.4.5.3 Duration

Although CPN notes have a specific written duration, the actual duration is usually shorter. The concept of duration is also harder to define for non-driven instruments – it is meaningless to ask the 'duration' of a cymbal crash, for example.

Offset times are probably not detected with as much resolution as onset times because note durations have less musical importance and note decays are usually less abrupt.

2.4.6 Acoustical illusions

2.4.6.1 Competing grouping mechanisms

There are many cases where the ear/brain can be misled. Diana Deutsch has studied grouping mechanisms extensively. One experiment[Deutsch 82] played the phrase shown in Figure 18, with the same timbres at hard left and hard right:-

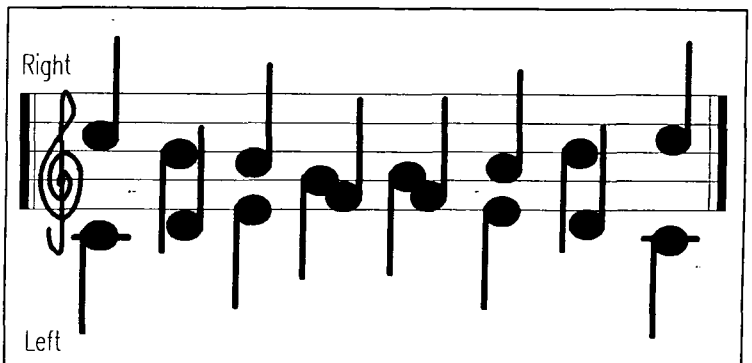


Figure 18 - Deutsch's example as played.

Subjects, however, tended to hear it as the *more musically plausible* phrase shown in Figure 19.

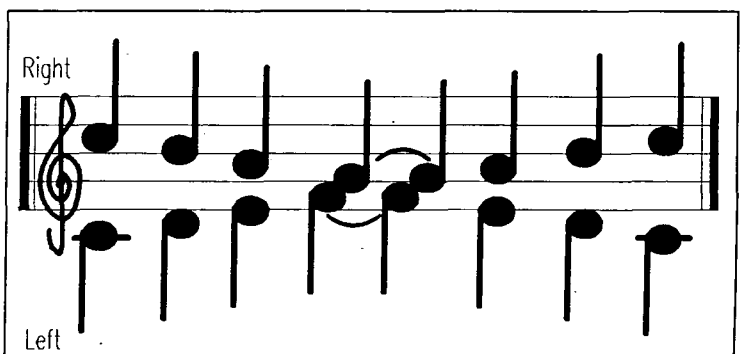


Figure 19 - Deutsch's example as heard.

In this case, melody consistency overrode positional information, but she showed that for another sequence, the reverse occurs. Butler showed that melody consistency can also override timbral differences.[Butler]

2.4.6.2 Inability to separate sources

Another case is when we hear a violin section playing in unison. Even if we know that there are sixteen players, we cannot distinguish them, and hear it as a single but very complex instrument.

2.4.6.3 The continuity illusion

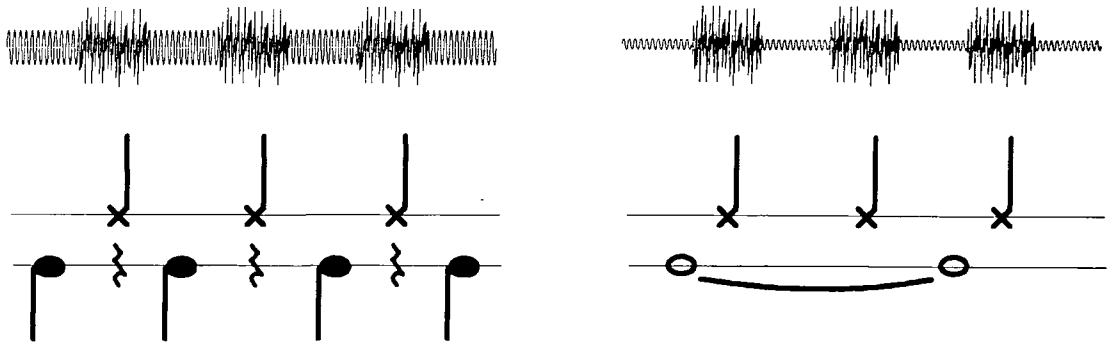


Figure 20 - The continuity illusion.

Another acoustical illusion is the *continuity illusion*.[Bregman 90, Rasch 78] This uses sounds consisting of 130 ms of a sine tone alternating with 130 ms of noise centred on the same frequency. If the noise energy is low, the listener hears the tone being *interrupted* by the noise, but when it is higher, the impression is that the noise has been *added* and that the tone is 'still there' beneath the noise. Figure 20 illustrates the continuity illusion. It is interesting to note that the illusion also occurs visually.

2.4.6.4 Effects of timbre on separation

Our separation of sounds, in a musical context, depends strongly on pitch and timbre. An interesting demonstration of this is given by Wessel.[Wessel 79] He played the sequence of tones shown in Figure 21. When the timbres are similar, an ascending B-E-A sequence is heard. When the timbres are dissimilar, two descending A-E-B sequences are heard.

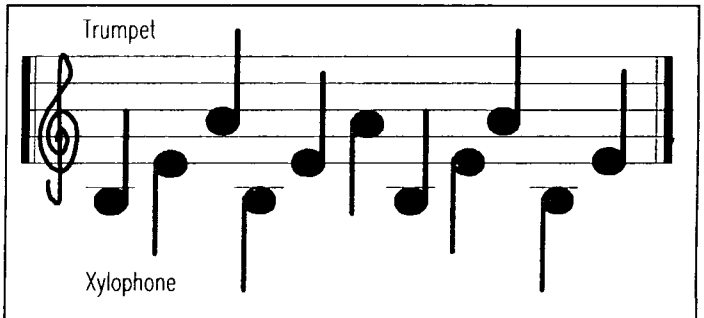


Figure 21 - Wessel's tone sequences.

Composers have been well aware of such effects. A single instrument can alternate between two melodic streams, or a single stream can be formed from several instruments, as shown schematically in Figure 22. This extract is from the second movement of Gunther Schuller's 'Symphony for Brass and Percussion'.

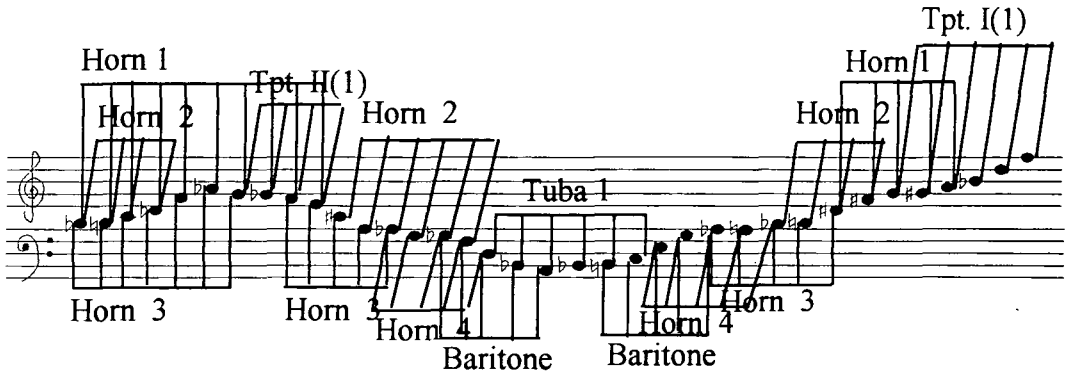


Figure 22 - Melodic streaming.

2.4.6.5 Discussion

In all these cases, there is a difference between the music *as played* and the music *as heard*. The question is:- which of these should a transcription system attempt to deduce? If we added a rule saying "If all the notes are C, E, and G, except for one G#, then change the G# to G", then we might correct an error in a simple piece but introduce an error to a piece that used a C/E/G/G# chord. Fortunately, the implementation described later is not yet able to carry out such high-level processing, so we can at least postpone answering the above question.

2.5 Summary

I have examined our low-level and mid-level perception of audio, and discussed our characterisation of notes and musical instruments. In the next chapter, I apply this to a discussion of analysis methods that can mimic some of the perception processes.

3. Computer analysis and synthesis

In this chapter I first examine several analysis and synthesis tasks that may be carried out automatically. I follow this by discussing how audio can be represented on a computer. Finally I outline possible routes to developing an analysis/resynthesis engine.

3.1 Analysis and resynthesis tasks

Below I discuss various analysis and synthesis tasks. While analysis and synthesis are often treated separately, they can also be viewed as complementary processes with similar aims.

3.1.1 Analysis without resynthesis

First I examine analysis tasks that do not require resynthesis. This classification is made because such applications are able to substantially reduce the amount of data.

3.1.1.1 Perception modelling

Research into psychoacoustics, fuelled by with the ability of computers to prepare more complex experiments on hearing, is making some progress, although it is widely recognised that the ear is a very complex organ. A system that can 'understand' sounds can be thought of as modelling our perception. The major difficulty in specifying a computer model lies in the fact that it is difficult to objectively describe what we hear, particularly when entirely new timbres are presented. Obviously, we cannot resynthesise a sound. Perception modelling thus must fall into the 'analysis without resynthesis' category.

Our recognition of musical events is restricted to how much we can take in in real time. An important distinction between modelling our perception and designing a transcription system lies in the fact that a computer can pass through the data more slowly (slow real time) or even in reverse (unreal time) to allow precise characterisation. However, this could not be deemed a valid model of the perceptual processes.

3.1.1.2 Note transcription

One task that musically-inclined humans can do relatively easily but computers cannot is to characterise the pitches, amplitudes, and durations of the notes in polyphonic music. For this, I will use the term *note transcription.* (The task of providing a *complete* description for exact resynthesis is discussed later and will be termed *full transcription.*) Here I discuss such systems that *do not completely characterise the input.* This is often done for expressive performance analysis or score derivation.

The resultant CPN notation or MIDI file only provides an approximate characterisation – "on the third beat of the eighth bar, the clarinet played an F# crotchet" or "at tick 9824, channel 7 plays note 66 for 100 ticks", not a complete specification sufficient for resynthesis. This is because timbral and control information cannot be written on paper. Figure 23, inspired by Tanguiane, illustrates.[Tanguiane 95]

Transcription has applications in musicological study and in auto-accompaniment systems for live performance. Previous work on transcription systems will be examined fully in the next chapter. Transcription systems are often designed for a particular instrument and/or for a particular piece. Given enough *a priori* knowledge of the signal's characteristics, an optimised design can be developed to look for features that we know should be there.

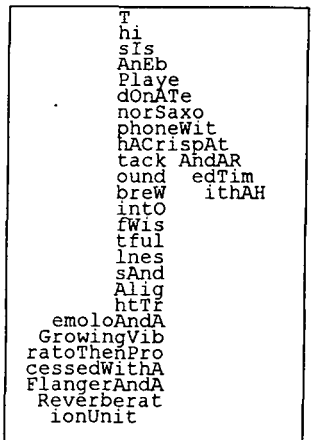


Figure 23 - What's in a note?

If the input is *known* to be monophonic, it is relatively straightforward to determine the fundamental frequency. For some instruments it can be assumed that the partials are close to being exactly harmonic. (See the earlier sections on frequency stretching and inharmonicity.)

However, transcription of polyphonic music is particularly difficult because before we can classify individual notes, we must in some way separate the individual sources. For this reason, some form of source separation is an important part of any polyphonic transcription engine.

3.1.1.3 Beat tracking

One application of analysis is in a field variously known as beat tracking, beat induction, or foot-tapping. Desain introduces many of these systems at the 1994 ICMC.[Desain 94] A practical application would be in automated mixing desks – for this, several pieces would be beat-tracked and rate-changed in order to synchronise their time structures. A logical extension of this would use chord induction to control pitch-shifting in order to also synchronise their temperaments.

3.1.1.4 Spectral analysis

Our inner ear converts the incoming pressure waveform into a frequency spectrum, and most systems for perceptual modelling or other audio analysis applications initially do something similar. Below I compare Fourier analysis with constant-Q transforms and a multirate scheme known as Octave Spectral Analysis.

3.1.1.4.1 Fourier analysis

3.1.1.4.1.1 Theory of Fourier Analysis

The main theorem of the Fourier Transform (FT) states that an arbitrary wave with period T can be rewritten as a sum of sinusoids at integrally related frequencies. The Discrete Fourier Transform (DFT) is basically the same for a discrete waveform. The Nyquist theorem qualifies this by stating that if the input contains frequencies up to f, then we must sample it at a frequency of 2×f if we are to correctly reconstruct the waveform. The Fast Fourier Transform (FFT) is a computationally efficient algorithm for calculating the DFT when the number of samples is a power of two.[Bracewell] For n samples, the DFT requires $O(N^2)$ calculations compared to $O(N.\log_2 N)$ for the FFT. Other integral factors can be

implemented in a similar manner through the use of the mixed-radix transform, albeit with smaller gains in efficiency.[21]

Although the FFT is widely used in fields from chemistry to seismology, it is ideal ONLY when the fundamental period is a known fixed power of 2. What if it isn't a power of 2? If it's still known, fixed, and integral, we can use the more computationally demanding DFT. What if it isn't known? If it's still fixed, then we can do one analysis to determine the frequency and a second to carry out the transform. What if it's not fixed? Then we can perform the transform over many shorter periods in each of which it can be assumed to be fixed. What if it has inharmonic partials? What if it's a musical noise such as a cymbal? What if we have the sum of many waves with different periods? What if we add the results of seventy-five people blowing into, hitting, bowing, plucking, or otherwise driving different non-linear air columns, membranes, bars, or strings? What if the timbres evolve continuously? What if they play with vibrato, tremolo, or rubato? What if there is noise, reverberation, tape flutter, or clipping? Clearly, the ideal conditions are never attained with real music. Some of these problems must be addressed in the design of the analysis system.

3.1.1.4.1.2 Resolution of Fourier analysis

The FFT is commonly used for frequency analysis where the range of interest covers a relatively small bandwidth, and we wish to determine the frequencies involved. However, music covers at least ten octaves, and the characteristic of interest is its pitch, which is related to log-frequency. With the FFT, the constant frequency resolution implies a widely varying pitch resolution. For example, a 1024-point FFT at a fixed sample rate of 44.1 kHz has bins 1-6 at F1, F2, C3, F3, A3, and C4 (middle C) to cover the entire range of the bass clef, as shown in Figure 24.

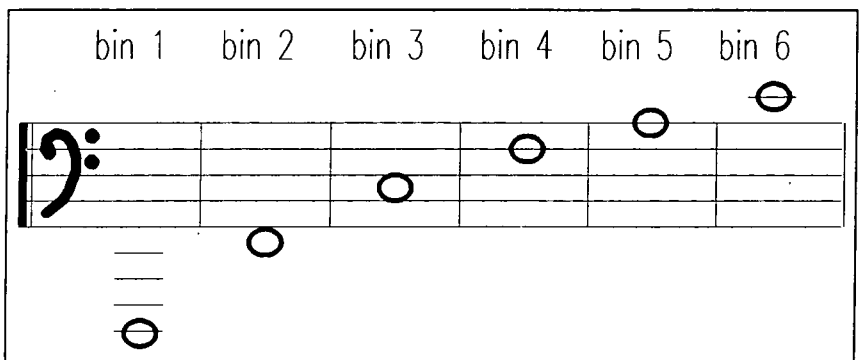


Figure 24 - Frequencies of bins 1-6 in 1024-point FFT.

Conversely, bins 256-512 give far more frequency resolution than is required, as they cover the single octave 11-22 kHz in steps of around 1/20 semitone. Linear and logarithmic spectra are illustrated in Figure 25.

---

[21] Even faster than the STFT is an approximate STFT. Nawab's scheme quantises the amplitudes to +1, 0, and -1 to derive an STFT with a 9-dB SNR.[Nawab] Hughes instead quantises the sine functions to these three values.[Hughes]

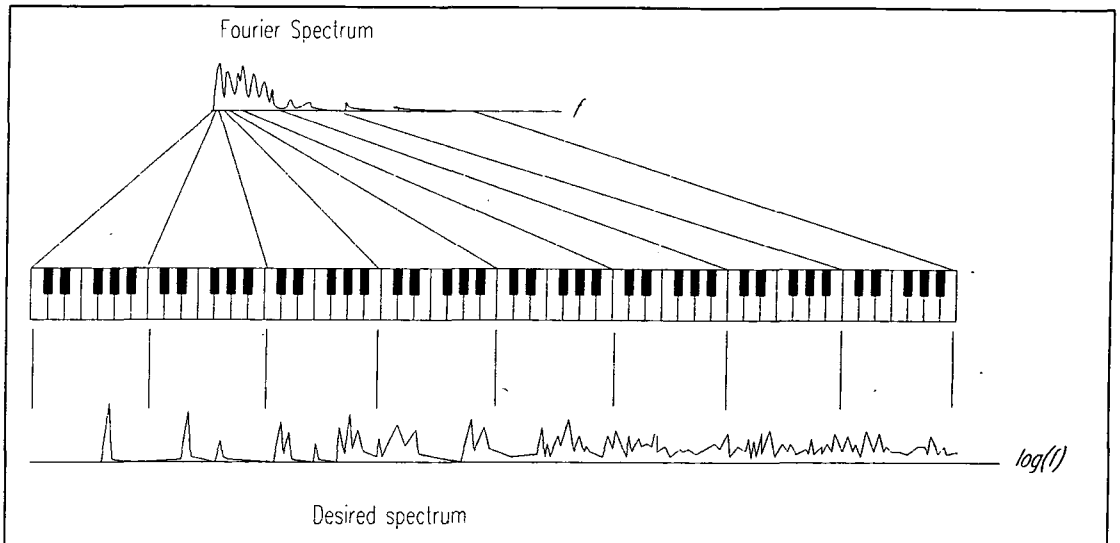


Figure 25 - Comparison of linear and logarithmic spectra.

One of the main problems with the FFT is that its resolution is a constant frequency, not a constant interval (log-frequency). This does not correspond to our own hearing. In the main part of our range of hearing, we can distinguish frequencies to an approximately constant resolution in log-frequency, meaning, *roughly*, that we can distinguish a low C from a low C# as easily as we can distinguish a high C from a high C#.[22] Another complication lies in the fact that we must hear a certain number of periods of a wave in order to determine its frequency to a given resolution, as indicated by Heisenberg's uncertainty principle. This makes it easier to distinguish pitch at higher frequencies *when the durations are the same*. This factor may partly contribute to the fact that we have poorer frequency discrimination at low frequencies.

Setting aside certain exceptions, it is fair to say that our frequency resolution is a constant interval, on the order of 10 cents (0.1 semitones) for notes of 'reasonable' length – a certain (and as yet undetermined) number of periods. This is justified by the fact that in conventional music, lower notes are typically longer. As a crude over-simplification, we might say that violins play semiquavers, violas play quavers, cellos play crotchets, and basses play minims. However, pitch is not frequency – pitch is determined by higher harmonics as well as the fundamental, so rapid bass notes can be articulated clearly. Figure 26 shows an excerpt from the double bass part of "The Magic Flute", shown as sounding pitch. Here the semiquavers have a length of around 90 ms, so the 55-Hz A is represented by around 5 periods. Such low notes would have to have a clear attack, though; one could not discern the pitches well if the same line were played slurred on a sine wave.

---

[22] This is not entirely accurate. As noted earlier, the critical bands in our ears are indeed larger at low frequencies. To counterbalance this, it is worth noting that the intervals used in the low register are generally larger; two bass instruments are rarely less than a perfect fifth apart. Our frequency resolution is also poorer at very high frequencies.

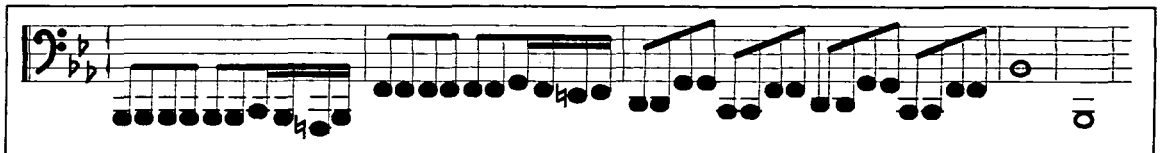


Figure 26 - Double bass line from overture to "The Magic Flute".

3.1.1.4.2 Constant-Q and gammatone filters

Constant-Q filters can be implemented by designing a filter (usually IIR) specifically for each analysis bin. This has been used in several analysis systems.[Kashino, Ellis 92b] However, such systems are generally designed for recognition but not resynthesis; the signal cannot be reconstructed as the impulse responses of the filters do not form an orthogonal set. Other researchers[BrownG 94a, Leman, Wöhrmann, Solbach 96a, Solbach 96b] have used gammatone or similar filters[PattersonR], which are a closer approximation to the ear's analysis, but in general these filters are not orthogonal and thus would not allow reconstruction of the input signal. (For a discussion of "Physiological vs. Functional Models", see Slaney's report on the Quebec CASA Workshop.[Slaney 95a])

3.1.1.4.3 Octave Spectral Analysis

A multirate system approximates the constant-Q transform by splitting the signal into individual octaves and analysing them separately. This is sometimes referred to as Octave Spectral Analysis.[Elliott] The basic principle is to use a half-band filter (also known as a Quadrature Mirror Filter[Vaidyanathan 87]) to extract the top octave, 11-22 kHz, and take its Fourier Transform. Since the remainder of the signal is below 11 kHz, we reduce the sampling rate to 22 kHz by only sending every second sample to the next stage. This is repeated as many times as necessary. A non-ideal filter will cause aliasing between adjacent octaves, but when certain conditions are met, we can guarantee that all such errors will cancel out exactly at the resynthesis stage.[Vaidyanathan 90]

3.1.1.4.4 Comparison of spectral analysis methods

With the FFT, increasing the FFT size allows more frequency resolution at the bass end, but decreasing it is the only way to get more time resolution at the top end. We would need the transform size to depend on the frequency for the filter to have a constant Q. With constant-Q transforms, it is possible to achieve this exactly, but such systems cannot give exact resynthesis. Multirate systems appear to be a viable compromise. The Q is *relatively* constant (it varies within a factor of 2), and exact resynthesis is possible.

A multirate system offers a more equitable allocation of the time-frequency bandwidth than a single-rate system, and it is expected that the benefits of more accurate analysis will outweigh the added

complexity. For these reasons, Octave Spectral Analysis appears to be the most promising option for spectral analysis.

3.1.1.5 Graphical display

The display of audio as one or two one-dimensional waveforms reveals little about its nature, expect perhaps its overall loudness. For a single tone, the timbre may be discernible, but only to those familiar with both the look and the sound of, say, a square wave. Even this is of limited use as timbres generally depend more on formants than on waveshapes.

A slightly better approach is to show the analytic signal, where we turn the input into two-dimensional data.[Justice] A single sinusoid can be viewed as the projection onto one dimension of a point on a circle, with a radius equal to the amplitude. The analogous process for an arbitrary waveform is known as the Hilbert Transform[23], and the two-dimensional space is known as Hilbert space. To do this, first take a set of short-time FFTs. Then, shift all the phases by 90 degrees[24], retransform to the time domain, and interleave these values with the original data. Finally, plot the N most recent line segments made from these pairs. (This process is actually carried out in five separate stages.) This gives a striking two-dimensional display, as in Figure 27, analogous to patterns drawn with the children's toy known as a spirograph.[25] For single tones, the effect of overtones is easy to see. If the tone has a strong sixth harmonic, as in the first example below, then the shape will have six clear 'petals'. If there is an inharmonic component, such as a component at 6.1 times the fundamental frequency, then the six petals will revolve slowly. This method of display is useful for single tones, but is of little use in representing many simultaneous notes. Another disadvantage is that for speed the current implementation draws straight lines between successive points in the x-y plane, whereas smooth interpolation would be more appropriate.

---

[23] The Hilbert Transform is equivalent to convolution with $1/(\pi x)$.

[24] An additional complication arises because we cannot determine the imaginary part of the DC component or the component at the Nyquist frequency. The latter will be negligible, but the former causes the pattern to jump erratically in one direction. At the boundary between successive time-frames, there will also be a small spike because of this.

[25] The mathematics of the spirograph – addition of phasors – is in fact identical to the mathematics of additive synthesis in the complex domain.

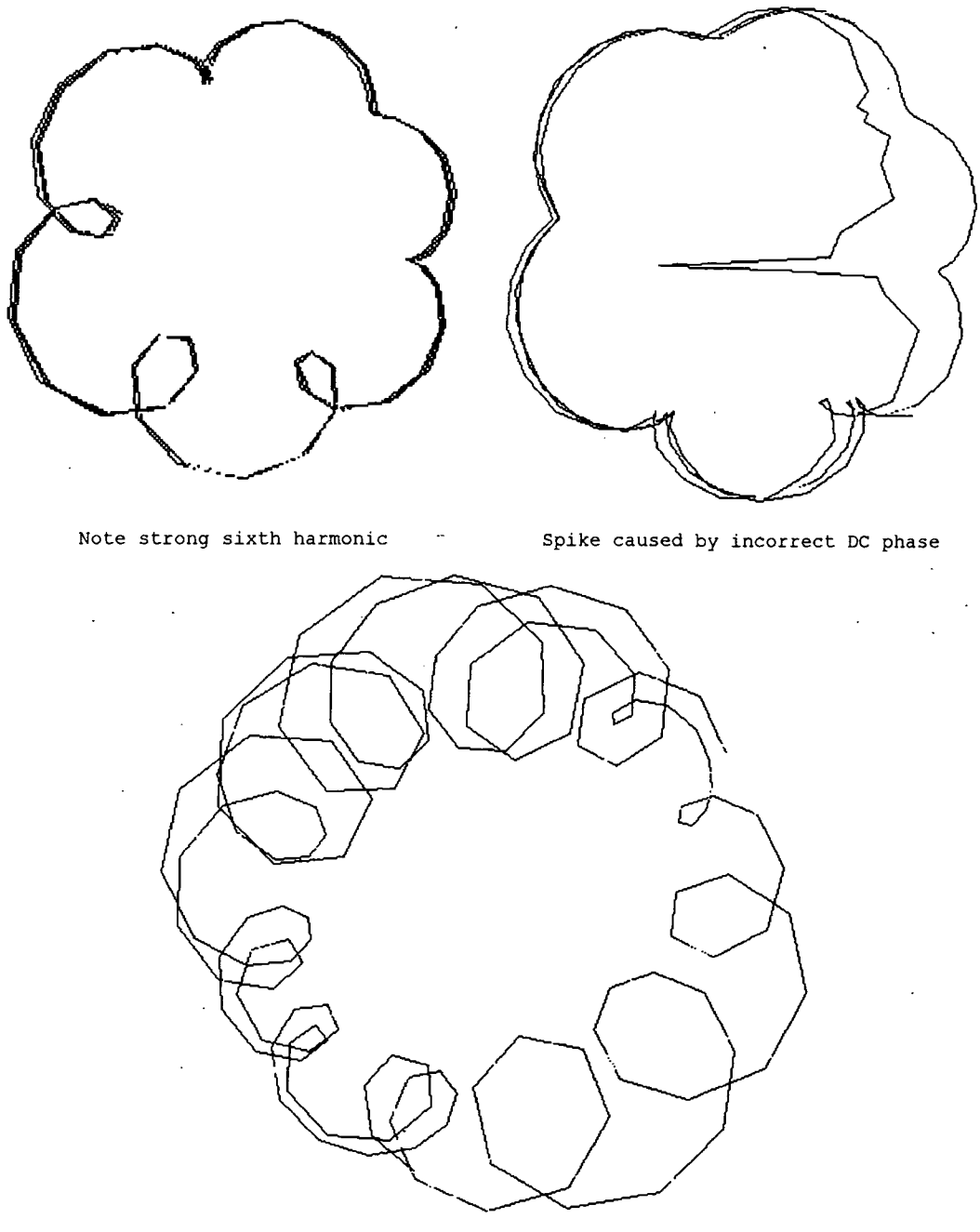


Figure 27 - Hilbert-space representation of waveforms.

For polyphonic music, this approach does not work due to the interference between notes, and a spectral method is called for. The *sonogram*[Ungvary] or *spectrogram* is a common way to show data derived from the FFT.

As an early experiment, a set of analysis and display routines based on the STFT were written for the PC. The results can be presented in many ways. Spectra are displayed by either assigning colours or

greyscales to different amplitude ranges, or by the contours of a spectrogram. It is difficult to pick out individual peaks in the standard spectrogram, so the axis was tilted as shown in Figure 28.

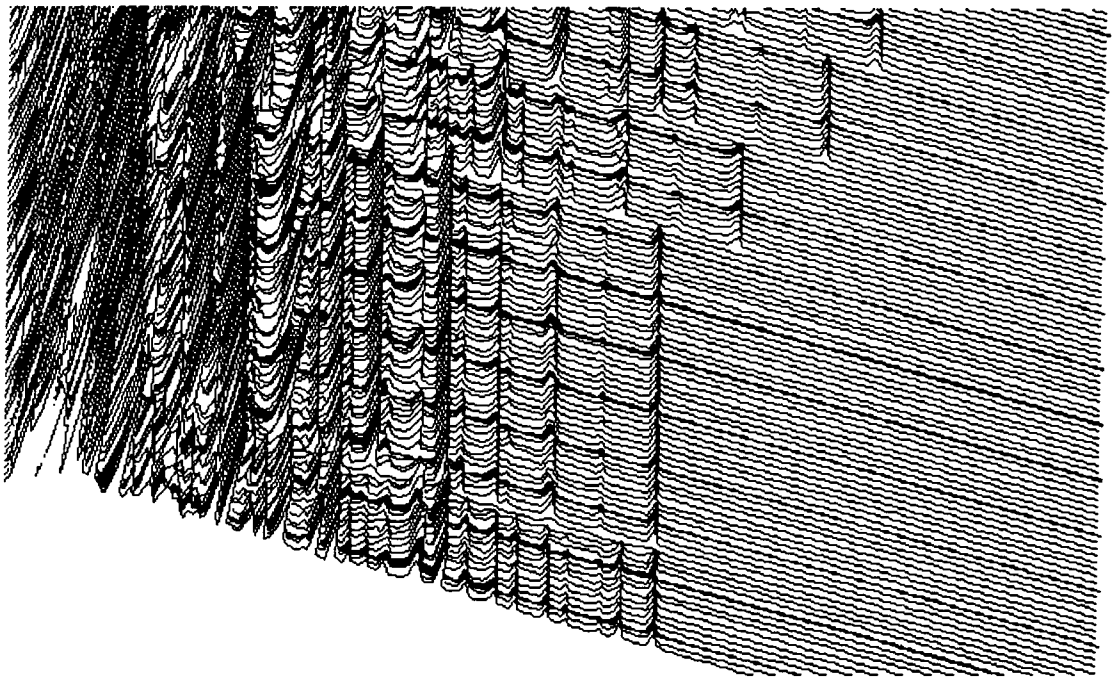


Figure 28 - Skewed spectrogram.

The spectrogram inherits and illustrates the disadvantages of the FFT, namely, its lack of a constant Q, or constant *pitch* resolution. Most of the information is gathered into the lowest bins. A similar tool[26] for displaying a multirate quasi-constant-Q spectrum will be developed and discussed in the next chapter.

Nevertheless neither the sonogram nor its multirate counterpart allow us to see concepts like trumpets and minor chords with the same ease that we can hear them, and can be described well using CPN. Acknowledging that some relevant information is not conveyed well by the sonogram, Helmuth describes a representation using five elements:- a sonogram, an amplitude representation, a traditional CPN stave for pitches, phrase marks, and text comments.[Helmuth 96]

Animated graphical display is also of primary importance for videos, although the aims here are often aesthetic rather than analytical. Several composers have examined ways of creating animations along with sound.[Pringle] Real-time performance is generally impossible and a second of output may require 10 to 1200 hours.[Bargar 92] This fact is grimly accepted by the graphics rendering community. It is also of interest to researchers to create animations directly *from* the sound. This can serve as documentation of electroacoustic music.

---

[26] The term *scaleogram* is often used for this.

3.1.2 Analysis and resynthesis

We now turn to systems that analyse data in order to resynthesise it.

3.1.2.1 Time shifting

A number of researchers have examined a problem that is deceptively straightforward to state but difficult to solve – *time shifting* or *timescale modification*. This means altering the temporal features of the sound without altering its frequency content. The fast-forward button on a CD player is a very crude example – it plays a little, then skips on. However, the artefacts are severe discontinuities. Dennis Gabor, pioneer in many fields, developed an optical device for time-stretching a film soundtrack. Manning describes another early tape-based device with four play heads on a rotating drum.[Manning] These techniques correspond to pitch-shifting by granulation, which is a popular choice.[Roucous, Jones 88, Lippe 93a, Truax 90, Truax 91, Truax 93, Truax 94, Di Scipio, Itagaki 96a] Researchers have also used the STFT[Portnoff, Settel]; the phase vocoder[Moorer 78, Dolson 86, Erbe], and wavelets.[Arfib 91, Ellis 92b] *Time stretching* is the most common case, where we wish to slow down a sound – very few researchers or composers have examined *time compression* as with fast-forwarding a CD. *Pitch shifting* is closely related, and refers to changing the frequency but not the pitches. This is equivalent to time shifting followed by resampling. Crude forms of this are sometimes implemented in children's toys and telephone voice changers as well as commercial effects devices and programs.[Prosoniq]

The task is difficult to perform without introducing artefacts because frequency is intimately dependent on time – we cannot distort the t axis whilst leaving the 1/t axis unchanged. I would go so far as to hypothesise that in many cases of time-stretching, the timbral detail that emerges may largely be the result of such artefacts, and perhaps not a magnification of the 'inner complexity' of the sound.

3.1.2.2 Timbral interpolation and cross synthesis

In searching for new sounds, composers may wish to create hybrid instruments from two others. This comes in two forms: *timbral interpolation* is forming a timbre between two other timbres[Moorer 77a, Lo, Haken 89], and *cross synthesis* is combining timbres in other ways, such as using the envelope of one instrument with the spectrum of another.[Kronland-Martinet 88, Settel, Horner 93, Rodet 94]

3.1.2.3 Compression

One 'musical engineering' challenge is straightforward to define: how much music can you fit into X megabytes of storage? With CD-quality coding, the answer is about 6 seconds per megabyte, or 1411200 bits/s. However, there is much less than 1 Mb of *perceptible* information in this 6 seconds. For transmission over networks, and for storage, we wish to minimise the amount of data without losing any information. There are three routes to this, as shown in the table below.

Type	*Perceptually different*	*Physically different*
Lossless	no	no
Near-lossless	no	yes
Lossy	no/yes	yes

Table 10 - Comparison of compression schemes.

Lossless compression of an *arbitrary* signal is, in theory, impossible. It is obvious that we cannot describe the 16 possible outcomes of tossing four different coins in three bits without losing information. The key is to realise that the input is not in fact arbitrary – we know that it is musical, and this points to a scheme by which the more like 'typical music' the input is, the more compactly it is encoded. This forces us to examine what the salient properties of music are, and to develop data structures into which the input should fit neatly.

Various compression schemes exist. The idea behind lossy compression is that it is possible to distort aspects of the waveform in ways that cannot or can hardly be heard. One lossy scheme is MPEG.[ISO, Hyun] The Motion Picture Engineering Group formed the MPEG audio standard, which exploits perceptual weaknesses such as our inability to clearly identify closely-spaced frequencies due to masking. In these cases, the output is not the same as the input, and the compression of the sounds may be traded off against the audibility of the differences. MPEG exists in various forms; 'CD-like' quality is achieved with 4:1 compression for audio layer 1, 6-8:1 for layer 2, and 12-14:1 for layer 3. These correspond to bit rates of 384, 256-192, and 128-112 kilobits per second.

The program SHORTEN by Tony Robinson can be run in either lossy or lossless modes. On average, it achieves 2:1 for 'strictly lossless' compression, and 4:1 for 'transparent' slightly lossy compression.[RobinsonA] Robinson estimates that 64 kbit/s for 'transparent' coding is achievable, a factor of 21.5:1. Another compression scheme is the Parallel Transform Method in which many compressors compete – this reportedly gives at least 2:1 compression with simple compressors.[Crandall] There is also DPCM and ADPCM, which capitalise on the high sample-to-sample correlation of musical audio. ADPCM gives lossy compression to 128 kbit/s for a signal band-limited to 15 kHz; this would translate to 188 kbit/s[27] for music at the CD sample rate.[Smyth, Bosi, Davidson]

Note that source separation is generally *not* part of a compression system – the compressor may separate different frequencies, but it has no concept of a frequency *belonging* to a particular note. A related application is denoising, where again complete source separation is not required.[BergerJ 94a, BergerJ 94b, BergerJ 94c, BergerJ 95, Settel]

[27] Here I ignore the fact that the compression may well be more efficient at higher sample rates.

3.1.2.4 Source separation

The "cocktail party effect" is a term for our ability to isolate and understand a single speaker when surrounded by other speakers.[Mitchell] This is an illustration of *source separation*. Music analysis is much more complex than speech analysis because music typically has an unknown and varying number of sources. We find it relatively easy to distinguish and identify these components, yet the task of separating the sum into its parts is horrendously complex.

In terms of information theory, when we combine signals, we invariably lose information. We might thus ask: if we can apparently separate one 16-bit signal into six 16-bit signals, where do the rest of the bits come from?

Source separation is an elusive goal in computer music research. It is mathematically impossible – we cannot derive a(t) from a(t)+b(t)+c(t)+d(t) – yet humans do it without conscious thought, and with no previous knowledge of the score or the instruments.

Ueda gives a good description of source separation, referring to it as blind decomposition. He observes that we can separate sources with a monaural signal, with unknown sounds, and with inharmonic timbres.[Ueda] He then presents results from a system that attempts to carry out blind decomposition of two sounds with no background noise. It does this by assuming (i) that all amplitude envelopes have the same shape, a questionable assumption, and (ii) that there is a frequency where the spectra do not overlap, which applies unless the notes are in unison.

Ueda's comments on monaural separation are accurate, and this work will largely examine the monaural case. Nevertheless, we are losing one potential cue for source separation[Bregman 89] – the pan position of each partial. For truly acoustic instruments, this is the same for each partial, although for heavily processed electroacoustic music, this is not necessarily true. Source separation may or may not be easier for stereo signals.

3.1.2.5 Full transcription

Note transcription in the sense of 'approximate characterisation' was discussed above. In this paragraph I use the term 'full transcription' to refer to the processes of source separation and complete characterisation of the input waveform, such as would allow resynthesis. As well as encoding, we wish to be able to carry out musically useful transformations on the data. Several systems for this task are reviewed in the next chapter. Almost all begin with one of the spectral analysis schemes described earlier.

This task is the most general. Full transcription includes source separation. As notes are the main mid-level entity in music, note transcription will be a by-product of full transcription. Moreover, if the representation is good, it is likely to permit compression. Indeed, the optimal representation can be said to be the one that allows the highest lossless compression.

Figure 29, after Risset[Risset 82], illustrates the framework for analysis and resynthesis.

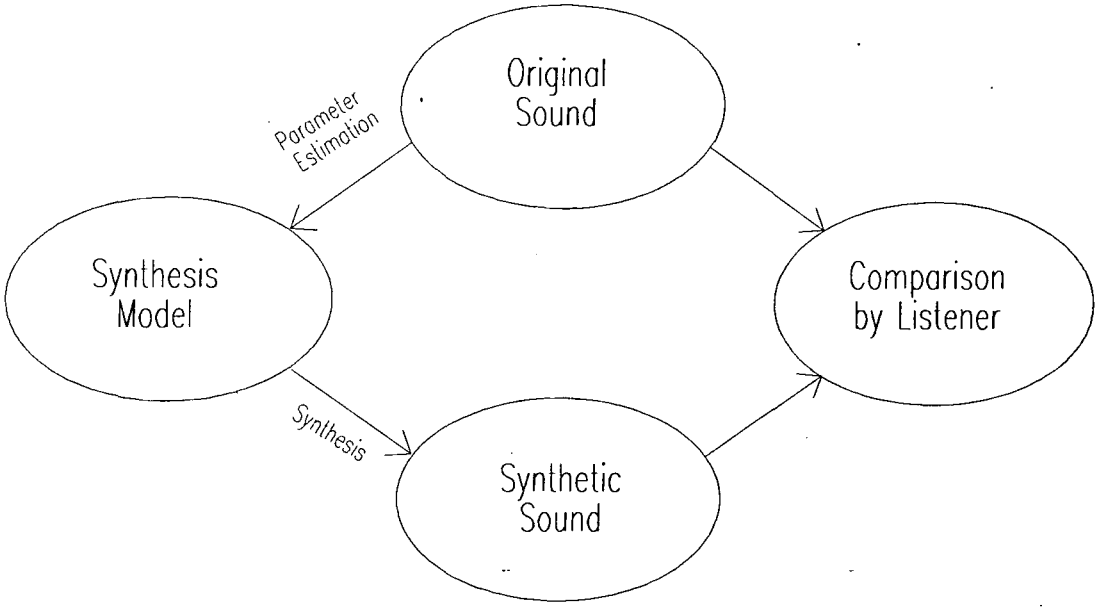


Figure 29 - The analysis/resynthesis process.

3.1.3 Comparison of analysis tasks

Task	*Known score*	*Known orchestra*	*Input*	*Source separation*	*Derive*	*Resynth.*
Spectral analysis	no	no	any	no	spectra	no
Perception modelling	no	no	any	YES	?	no
Mono note transcription	YES/no	YES/no	mono	no	midi	no
Poly note transcription	YES/no	YES/no	any	YES	midi	no
Time shifting	no	no	any	no	wave	YES
Compression	no	no	any	no	binary	YES
Source separation	no	no	poly	YES	waves	YES
Full transcription	no	no	any	YES	synth params	YES

Table 11 - Comparison of analysis tasks.

The above table summarises the differences between the main analysis tasks. Transcription is not always intended for, and does not necessarily permit, resynthesis. Various types of transcription exist, and of these, some depend on other assumptions about the input. Compression implies that resynthesis will be used, and that no assumptions will be made about the input. Lossless compression thus overlaps with the last form of transcription. The difference is that in the first, the primary aim is data size, and the data will not be processed. In analysis/resynthesis, however, we are hoping that musically relevant transformations can be carried out in the intermediate domain.

3.1.4 Specification of our task

The system that will be developed in the next chapters is designed for analysis and resynthesis – it is intended as a 'full transcription' system, with resynthesis as an eventual goal. 'Note transcription' is a useful side-effect that allows judgement of the accuracy. Compression *may* be a side result of this process, but this is not its primary use.

3.2 Representations for music

The following section compares various representations of musical data, and discusses their differing advantages and disadvantages for analysis and synthesis.

3.2.1 Wave-based representations

One of the few things that can be said with complete certainty about musical audio is that it is a continuous signal. A few representations are inherently continuous, but many more are discrete. Other methods make use of 'frames' of information.

3.2.1.1 Continuous representations

The concept of continuousness in a waveform can be modelled by fitting the waveform to a series of polynomials. One example is the use of Bezier curves in graphics fonts. These have the advantages of being compact and instantly scalable. Compactness is of course desirable, and the scalability is most attractive for time-stretching. It is also easier to add polynomials than to add sinusoids, for example. However, polynomials inevitably tend to $\pm\infty$ as $t \rightarrow \pm\infty$, and as a result are not well suited to modelling periodic functions. Piecewise polynomials are advocated by Hung, who uses them for analysis/resynthesis and shows that 'high subjective similarity' is obtained while giving compression to 0.1-0.2 times the size.[Hung]

3.2.1.2 Discrete representations

3.2.1.2.1 Pulse Code Modulation (PCM)

The most obvious, and the most common, digital representation of a continuous wave is to sample it at equally-spaced time intervals. Nyquist's theorem tells us that when the maximum frequency is f Hz, we can represent it by samples at a rate of $2 \times f$ Hz.

In practice, the values are quantised to a discrete set. The values may be linearly or logarithmically spaced. Linear encoding is simpler, and is the standard for CDs. The signal-to-noise ratio (SNR) for linear encoding is given by:-

$$SNR \leq 4.8 + 6.0 * Nbits$$

In theory 16-bit encoding gives an SNR of 100.8 dB. However, as the noise floor is fixed, the SNR depends on the overall amplitude. In pianissimo passages the SNR is much poorer. Logarithmic

encoding, or companding, gives an approximately constant SNR, and higher effective fidelity per bit, but is more complex and less widely supported.

3.2.1.2.2 Irregular samples

It is also possible to use unequally-spaced samples to record a wave. This in principle allows the data rate to be lowered to twice the *local* maximum frequency. However, the fact that the times of the samples must also be encoded in the output waveform is likely to outweigh any compression, and both processing and playback are considerably more complex.

3.2.1.2.3 Predictive coding schemes

A sample of a continuous waveform inevitably has a high autocorrelation at small delays, and in music, the high-frequency content is much lower than the low-frequency content. Various statistical coding schemes exploit this to allow compression. Delta modulation, DPCM (Differential PCM), and ADPCM (Adaptive Differential PCM)[Smyth] use a prediction method, and send the difference between the predicted and the actual values.

3.2.1.3 Frame-based representation

Frame-based representations divide the signal into frames, often of size 2^N. It should be noted immediately that this is an artificial division and has no correlation with any periodicity in the input.

3.2.1.3.1 Short-Time Fourier Transform

The Fast Fourier Transform[Bracewell] presents a feasible scheme for encoding music, by dividing the signal into blocks of size 2^N and recording the FFT of each. Of course, this gives no compression. Compression can be achieved only by missing out the lowest-level bins, but this adds the overhead of coding which bins are present. Often, the spectra are converted to a set of linked amplitude-frequency envelopes, and straight-line interpolation is used to compress the data, but this adds distortion.

The STFT is popular as a starting point in analysis leading to additive or Fourier synthesis. However, the time/frequency resolution is less than ideal, as will be discussed later. Another objection is that if the source contains a single sine wave at 123.456 Hz, the Fourier transform cannot represent it compactly. The FFT permits multiples of $M \times 44100/2^N$, but no other frequency can be represented; it in principle requires all frequencies in order to reconstruct one of the non-chosen frequencies. In much the same way, an arbitrary time cannot be specified – it cannot tell us what happened at sample 1000, it can only report on the periods 896-1023 and 1024-1152.

3.2.1.3.2 Linear Predictive Coding

LPC breaks the input into frames and models each frame by an n-tap filter excited either by noise or pulses.[Markel] This is designed primarily for speech applications, where it is expected that there is a single signal in the input, and that exact reconstruction is not necessary. It is unlikely that this method would work effectively with a more complex signal such as polyphonic music.

3.2.2 Event-based representations

Many other coding schemes make use of *events*. An event "happens" *at* a single point in time, rather than existing *over* a period of time. They encapsulate information regarding the sound between T_1 and T_2 into an entity at the start point T_1. For example, a MIDI note at time 1000 might represent a waveform between times 1000 and 1100. In many cases it is indeed valid to suggest that every note has a start and an end, but this is not necessarily the case. On a non-driven instrument, such as a free harp string, a note has no offset. It will die towards zero, but the point to which we assign the end depends on the available resolution and/or the limit of audibility. Similarly, if an instrument fades in gradually, there will be to the listener an increasing certainty that the note is there, but there will be no point when the note started. This is also the case when an instrument is reversed, the reversed cymbal being the best example.

When there is little change in the characteristics of the note over its duration, events are a particularly efficient form of coding. On non-driven instruments (e.g. plucked strings, percussion), the initial energy dissipates; there is generally no control during the note, except perhaps for stopping it. However, on driven instruments (bowed and blown instruments, voice, tape, electric, electronic), notes can be sustained for long durations, and there may be a high amount of control information as the note evolves. A single note might be a ten-minute performance on a didgeridoo. In such cases, events are less effective, as they must be accompanied by a stream of control information. The size of this control information is one of the main concerns of this research.

The control information may be a continuous variable (such as slide position, bow angle, or tongue position), and this implies that the bandwidth of control information will be band-limited. The bandwidth of control information for acoustic instruments is unknown, but we can argue that the maximum frequency of one parameter cannot exceed half of the fundamental frequency. It is impossible to impose a 30-Hz vibrato on a 40-Hz fundamental as this would be perceived as a different (FM) timbre rather than a control envelope. However, there may be several independent controls – the control information is not one-dimensional.

In most electronic instruments, the control information is digital and thus discrete, and falls at times dictated by the MIDI clock, the Csound[Vercoe 90, Vercoe 93] control rate (typically 20 ms), or an unknown parameter inside a proprietary chip. (Arguably, acoustic instruments can have discrete controls – violin trills, xylophone rolls, and guitar hammer-ons being examples. All of these, however, can also be viewed as successions of notes rather than single notes.)

The assertion that music is necessarily organised in terms of discrete events bears further scrutiny. It is certainly possible to compose music that consists primarily of gradual textural changes, with no sensations of onsets and offsets. However, it is fair to say that this music forms a small minority.

There are several reasons for attempting to convert a waveform into an event-based representation. First, composers generally wish to create music as parallel streams of events.[28] Second, we perceive music as events, and higher abstractions such as harmonies, melodies, and rhythms are defined in terms of events. Third, events allow us to separate the time-domain information from the frequency-domain information, for example, to change the pitch independently of the duration or vice versa.

3.2.3 Wavelet representations

Much recent research has revolved around the wavelet transform.[Daubechies 88a] A wavelet is a short wave, rather like a grain in granular synthesis. It is a short continuous function, specified by a small number of parameters. Wavelets are explored more fully in a later chapter.

3.2.4 Comparison of representations

Since the invention of the wax cylinder, it has been conventional to regard music simply as one-dimensional audio waveforms, to be recorded, stored, and reproduced with as little distortion as possible. Both analogue and digital can now offer sufficient fidelity. The waveform representation has the important characteristic of almost-complete *generality*, in that any band-limited waveform can be encoded to an arbitrary resolution determined by the effects of quantisation noise. However this generality comes at the expense of size – it requires around ten megabytes per minute for CD-quality sound. This typically places high demands on both the processor and the disk. Second, the syntax cannot be related to the semantics, in that the string of numbers tells us nothing about its meaning – i.e. the way we will perceive it. The waveform representation is much more suited to the media than to the human.

An event-based representation, such as the MIDI file, the printed score, and the player piano roll, is at the other end of the scale. It does not allow generality, but has the equally important attribute of *compactness*. It also scores highly for *intuitiveness*, as its parameters – note number, loudness, and duration – indicate the method of producing the sound (in the case of traditional music), and (arguably) the parameters by which we perceive it. Another advantage of such representations is their *parallelism* – we record what each instrument does, rather than the sum of the whole ensemble. (With MIDI, a major complaint has been its limit of 16 channels.) Parallelism permits detailed editing at the symbolic level. However, the global timbres of each instrument must be defined separately, and the complexity and controllability of each note are limited by the capabilities of the target synthesiser.

There are many alternatives between these two extremes. When the aim is solely the creation of new music, most musicians are happy to accept the restrictions of the chosen synthesis method, and to adjust the performance parameters until the results sound good. However, this cannot be used for processing

---

[28] As early as 1977, Moorer ascribes the preference of music languages (acoustic compilers) over general-purpose languages to the desire for parallelism.[Moorer 77a]

audio, as there is generally no way to map an input waveform onto the control parameters. Synthesis should be viewed in a broader context, as one part of the analysis-transformation-resynthesis paradigm.

The optimal representation for music would be *general, compact, parallel, and intuitive*. Generality means that we must be able to encode any input, yet lossless compression of an arbitrary signal is theoretically impossible. A central hypothesis of this research is that any *musical* waveform contains sufficient redundancy to be coded more efficiently. To exploit this redundancy, however, we must first break the waveform into its constituent musical entities, i.e. we must carry out source separation.

	Wave-based	Event-based	Wavelet-based
General	YES	no	YES
Parallel	no	YES	YES
Compact	no	YES	YES ?
Intuitive	no	YES	YES ?

Table 12 - Comparison of representations.

As shown in the table above, wave-based representations' only benefit is their generality. Representations based on events offer parallelism, compactness, and intuitiveness, but most of the methods described are unsuitable for analysis-driven resynthesis. The most promising options appear to be additive synthesis, parallel wavetables, and wavelets.

Wiggins discusses representations, concentrating on note-level issues rather than waveforms.[Wiggins] He discusses the need for a representation to permit a range of structures above notes, such as chords, rhythms, and trills. Dannenberg also examines representations in some detail.[Dannenberg 93a]

3.3 Choice of paradigm

It is worth examining existing synthesis schemes to determine their applicability to analysis and resynthesis.

Composers are free to use any method available to create sound. Since the composition process entails turning a small amount of data into a large amount of data, much research has gone into developing ways of creating new and interesting sounds given a finite and insufficient amount of computational power. Julius Smith gives a good overview of synthesis techniques.[SmithJ 91] In some cases, synthesis is done without regard to laws governing vibration of real objects. There is often no analysis scheme that corresponds to the synthesis method.

The following are Jaffe's ten criteria for evaluating synthesis techniques[29], with my interpretations added in italics.[Jaffe]

[29] One criteria Jaffe misses is "Can timbres be combined?" Composers wish to use *timbral interpolation* or *cross synthesis*, combining the characteristics of two instruments.

1)	How intuitive are the parameters?	*Is there a brightness knob?*
2)	How perceptible are the parameter changes?	*Can I hear what each knob does?*
3)	How physical are the parameters?	*Can I map things like 'bow speed'?*
4)	How well behaved are the parameters?	*Does a small tweak cause a small change?*
5)	How robust is the sound's identity?	*Can I make all the sounds of an instrument?*
6)	How efficient is the algorithm?	*Is it computationally costly?*
7)	How sparse is the control stream?	*How low is the bandwidth?*
8)	What classes of sounds can be represented?	*Is it general?*
9)	What is the smallest possible latency?	*How fast is it?*
10)	Do analysis tools exist?	*Can I input any sound?*

While all of these are valuable properties, it is the last of these that is the most critical factor in considering their suitability. Below I examine the applicability of some existing synthesis methods to analysis and resynthesis.

3.3.1 Additive synthesis

One traditional approach to synthesising sound has been to treat individual notes as being either strictly periodic or quasi-periodic, and to generate each partial separately. I refer to this as additive synthesis, although it should be noted the term is sometimes used to mean *any* method that constructs sound by adding things (such as granular sampling). Those who favour the latter terminology use the term *Additive Sine Wave Synthesis (ASWS).*[Houghton] Others prefer the slightly misleading term Fourier Synthesis.

If the pitch is known, then the Fourier Transform can be used to give the amplitude envelopes of each harmonic. In some schemes, partials are assumed to be harmonic, which allows computational efficiencies. In other schemes, harmonics need not be at multiples of the fundamental frequency, which means we must also record the frequency envelope of each partial.

The analysis can be done using Fourier or other spectral analysis *if* each note is available individually, but in polyphonic music, the spectra of individual notes interfere with each other, and thus the partials cannot be determined easily. The implementation of Fourier analysis is discussed in more detail in a later section.

3.3.1.1 Number of sines

Additive synthesis is computationally expensive. The amount of computation depends on the polyphony and on the number of partials needed per note. Samson quotes 20 to 30 partials for a bowed string; Moorer uses 21 for cello, clarinet, and trumpet tones; Wessel uses 25; Haken quotes 20 to 80, Houghton uses 64, and Freed says a hundred or more are needed for low piano tones.[Samson, Moorer 77a, Wessel 78, Haken 92, Houghton, Freed 93a] This implies that additive synthesis of an ensemble could involve thousands of partials. This means serious processing power is needed – approaches include Cray

supercomputers[Kriese] and custom-built VLSI.[Houghton, Phillips 94, Phillips 96] Possible optimisations include CORDIC operations[Hu, Phillips 94, Phillips 96] or a multirate approach.[Phillips 94, Phillips 96, Nunn 94]

It is worth noting that no current commercially-available synthesiser is based on additive synthesis.[30] As well as computational restrictions, this may also be due to the lack of easily-tweakable parameters. If we wish to increase the 'brightness' of a note, we must update all of the partial amplitudes.

3.3.1.2 Synthesising sines

There are two ways to generate sines. One is to use a recursive oscillator, and the other, a more popular choice, is to use a lookup table.[MooreF 90, Freed 93a] The SNR of a lookup table is a function of the resolution in the magnitude and time domains.[MooreF 77] Listening tests show that an isolated sine from a 4k by 12-bit lookup table is heard as being as good as one from a 64k by 16-bit table.[Snell] However, Jansen recommends a larger table to minimise distortion when there are many close sinusoids.[Jansen 91]

3.3.1.3 Hardware-based approaches

The 'strong-arm' method of generating many partials is to use powerful hardware. Cor Jansen presents a scheme using a transputer-based architecture that can generate 10000 partials at 44.1 kHz, 625 on each of 16 cards.[Jansen 91, Jansen 92] It can also generate short bursts of 'noise' using many sinusoids 16 Hz apart, although he notes that for longer periods the sines should be more closely spaced. Transputer-based parallel architectures are also under study in this research group. Takebumi Itagaki presents an implementation for 27 notes of 24 partials on a ternary tree of T800 transputers.[Itagaki 94, Itagaki 95a] Des Phillips defends the hardware approach to synthesis, and outlines a CORDIC coprocessor to do the number-crunching.[Phillips 94, Phillips 96]

Haken implements 100 partials on the CERL Platypus system, and gives (debatable) informal results suggesting that no more than 75 partials are needed.[Haken 92] Houghton developed an ASIC (application-specific integrated circuit) for sine wave synthesis, and shows a prototype where 127 sinusoids can be generated on a card that plugs into a PC.[Houghton] Di Giugno presented a chip for ASWS with 256 oscillators.[Di Giugno]

3.3.1.4 IFFT

Additive synthesis can also be carried out by mapping sinusoids onto an STFT and then using the Inverse FFT.[Depalle 90, Depalle 93, Rodet 92a, Rodet 92c, Freed 93a, Freed 93b] Freed shows that this can be many times more efficient and could allow several hundred partials on a 'desktop computer', which is a rather

---

[30] Older synthesisers that used additive synthesis include the Casio SK-1 and FZ-10, the Kawai K-1, K-5, and K-5000, the Fairlight CMI and Fairlight II, the Axcel resynthesizer, the Kurzweil K-150-FS, the Lyre Fourier Digital Synthesizer, the Synclavier, the Synergy GDS, the Slave 32, the Synergy GDS, the Seiko DS-250, the RMI harmonic synthesiser, the Akai AX-80, the Oscar-1, and the Korg DW-6000.

modest description of his Silicon Graphics Indigo workstation. Rodet quotes 300 sinusoids on the same machine.

3.3.1.5 Envelopes

Most researchers using additive synthesis come up against the same problem – even when we have the envelope of each harmonic, this is still a large amount of data. Sometimes a block-based approach is used; as Dannenberg points out, this adds efficiency at the expense of accuracy.[Dannenberg 92]

To reduce the amount of control data, piecewise linear approximation (PLA) is often used, giving interpolation between several[31] 'breakpoints' in the envelope.[Risset 69, Beauchamp, Grey 75, Grey 77b, Wessel 78, Strawn, Feiten 90, Jansen 91, Kriese, Horner 95] Horner gives an overview or many of these.[Horner 96] This largely achieves its aim, giving compression from 43:1[Risset 69, Moorer 77a] to 100:1[Serra 90], but also introduces artefacts – each sharp corner in the envelope causes a transient with a power spectrum falling at 12 dB/octave. While experiments seem to indicate that much data can indeed be discarded[Risset 69, Beauchamp, Grey 75, Grey 77b], Strawn notes that "there is still no definitive answer to the question of how much data can be omitted without changing the tone significantly". Additive synthesis methods' claim to reproduce the exact nuances of a sound is compromised when such approximations are made.

Assuming linear interpolation of amplitude *and* frequency, and the use of a lookup table for sine generation, the following table shows how many operations are required per partial per sample.[Freed 93a]

Operation	*Additions*	*Multiplications*	*Modulo*	*Lookup*
amplitude interpolation	1 fp			
frequency interpolation	2			
sine evaluation			1	1
output	1 fp	1 fp		

Table 13 - Operations for additive synthesis of one sine.

3.3.1.6 Group additive synthesis

A change in one actual controller, such as tongue-palette distance, will affect all of the harmonics, often in similar ways. One way to reduce the high computational cost of additive is to use fewer envelopes than partials, and interpolate the missing ones. This is known as *group additive synthesis*.[Kleczkowski, Eaglestone] The problem of determining the simplified set of envelopes has been approached using genetic algorithms and principal component analysis[Horner 95], multidimensional scaling[Grey 77a, De Poli 93], neural networks[De Poli 93, Mourjopoulos, Feiten 91, Kohonen], and wavelets.[Kronland-Martinet 93] Horner shows that trumpet and pipa tones could be modelled by four different amplitude envelopes and an erhu by three,

---

[31] The number of breakpoints required varies from nine (piano and guitar) to twelve (trumpet).[Horner 96] In the best conditions listeners needed 12, 20, and 25 breakpoints for indistinguishability. The timbre of a didgeridoo will be analysed later; a long note would require many more breakpoints.

with most listeners unable to distinguish them from real tones. Kronland-Martinet also successfully used six master envelopes to represent 32 partials of a trumpet sound.

As well as sharing amplitude envelopes, it is also possible to share frequency movement envelopes between partials. On real notes, each partial will have the same rate of vibrato, and this is known to be an important factor in fusion into a note.

It should, however, be remembered that these assumptions may not apply to non-acoustic instruments and thus detract from the generality of an analysis method. It is easy to form artificial tones that do not conform to a model's expectations – for example a tone whose partials had different vibratos and tremolos, and stereo locations.

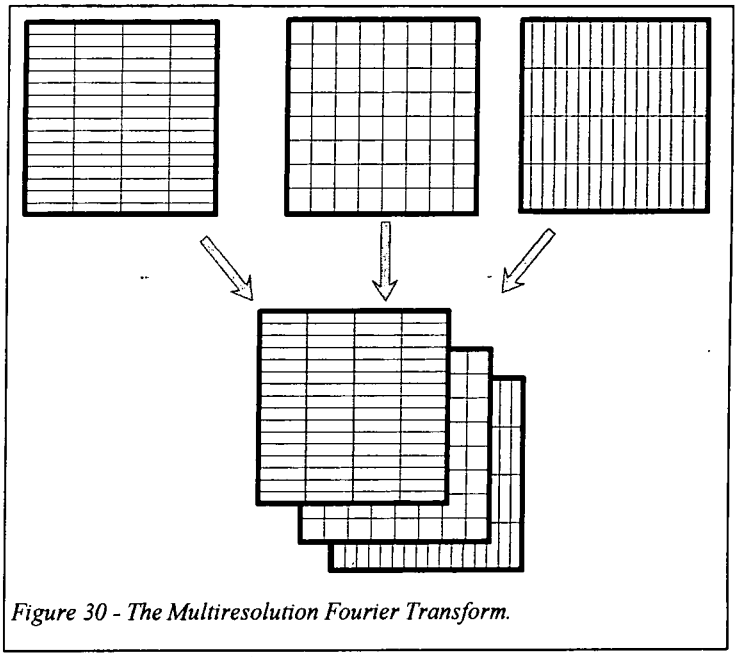


Figure 30 - The Multiresolution Fourier Transform.

3.3.1.7 Alternatives to FFTs

Some analysis methods use methods similar to, but not, the FFT. This includes the Modified Moving Window Method (MMWM), invented by Kodera.[Kodera 76, Kodera 78, Auger, Höldrich 95] Another variant is the MFT (Multiresolution FT)[Calway, Pearson 91], which uses STFTs of several lengths simultaneously. This is shown in Figure 30.

Other techniques include the *phase vocoder*, which is often used for speech[Dolson 91], and linear prediction.[Rabiner 78, Markel] The cost depends on the number of partials or filter taps. This is equivalent to the number of concurrent partials, which is low for speech but high for music. Speech research has also employed *cepstral* methods[32], which are defined using the 'spectrum' of the magnitude spectrum of a signal.[Noll, Pabon 94b, Boatin] An interesting approach using higher-order statistics of the signal known as *polyspectra* is also outlined by Dubnov.[Dubnov]

3.3.2 Amplitude modulation

AM, in its simplest form, produces three spectral components for the computational cost of two.[Dueşsenberry] However, we do not have independent control over the two sidebands so there is no computational gain if generality is required. Ring modulation (RM) is a closely related technique, more

[32] Cepstral techniques have given the language the words cepstrum, quefrency, and saphe.

technically called double-sideband (DSB) suppressed carrier modulation.[33] There is no corresponding analysis method, although Delprat illustrates how AM parameters can be derived from additive synthesis parameters.[Delprat 90]

3.3.3 Frequency modulation and phase modulation

The distinction between FM and PM is primarily a mathematical technicality – many "FM synthesisers", including the Yamaha DX7[Chowning 73, Chowning 86a] and SY77[Yamaha], actually implement phase modulation, and the term FM is often used to mean FM *or* PM.[34] Both can create a rich spectrum from a small number of controls. FM, needing two modulo operations, two table lookups, two multiplications, and one addition per output sample. Discrete closed-form summation formulae are a generalisation of this technique.[Moorer 76, Moorer 77a, MooreF 90]

FM is a prime example of real-time synthesis with limited computational resources. Since real-time additive synthesis was not possible, Chowning then asked "what *can* be produced in real time?", and found a computationally efficient way to make *something*. It was then realised that the something was a new and unexplored range of timbres, and the Yamaha DX7 became hugely popular. Naturally, this popularity led to FM sounds becoming more commonplace – the sounds are now almost passé. FM sounds correspond to an electronic rather than a physical process, and recreate little of the realism of acoustic instruments.

As with AM, there is no directly analogous analysis technique, but Delprat's analysis also succeeds in extracting PM parameters.[Delprat 90]

3.3.4 Physical modelling

Physical modelling (confusingly also abbreviated to PM) has attracted much attention recently.[SmithJ 92, Rodet 93, Jánosy, Szilas] Computer models of physical objects can be implemented using waveguides, and these can be shown to have many of the quirks of real instruments. Models exist for stringed instruments[Chafe 91, SmithJ 92, SmithJ 93, Karjalainen 93, Karjalainen 96, Kurz], brass instruments[Rodet 96], woodwind[Cook 88, Välimäki 93, Välimäki 96, Verge 96, Scavone], voices[Cook 90, Sawada], and percussion.[Van Duyne 93, Van Duyne 96, Fontana, Cook 96] Transitions between notes, for example, are realistically synthesised. Another advantage is that they allow easier interfacing to physical controllers. However, the method is again synthesis-oriented, and is only suitable for instruments for which a physical model has been derived. There is no way to transcribe an arbitrary waveform into a physical model; physical modelling largely derives its parameter settings by the 'tweak-it-and-listen' approach.

---

[33] The name is derived from its implementation using a ring of diodes.

[34] The output of modulation is $\cos(\omega.t + z)$. If z is proportional to the modulating signal, we get phase modulation. If z is proportional to the derivative of the modulating signal, we get frequency modulation.[Schottstaedt]

3.3.5 Waveshaping

If we generate a sine and apply a non-linear function to it, other frequencies are generated at multiples of the fundamental. This is known as waveshaping. Chebyshev polynomials can be used to derive the waveshaping function for a given spectrum. This does not reproduce phase information and cannot give inharmonic timbres. Delprat's analysis has been extended to deriving non-linear waveshaping functions.[Kronland-Martinet 93]

3.3.6 Subtractive synthesis

With subtractive synthesis, we take a broadband source – often noise or impulses – and apply filters to shape the spectrum as desired. This is suited to approximate reconstruction but not to exact reconstruction.

3.3.7 Chaos

Chaos, or non-linear dynamics, is potentially an attractive way to produce musically interesting sounds.[Rodet 92b, Pressing 93b, Degazio, Radunskaya, Milicevic] However, again there is no corresponding analysis method, and by definition the calculations are unstable.

3.3.8 Wavetables and sampling

Wavetable synthesis is another common method of synthesis[Mathews 69], and is implemented on many synthesisers.[Yamaha] There are three categories of wavetable synthesis, depending on the size of the data block. At one extreme is sampling, where it is possible, and fairly common, to use very long sample loops containing many notes on many instruments. In the middle are samplers and sample-based synthesisers, where each sample represents a single note. However, the spectrum of an instrument generally depends on its pitch, requiring us to use many samples to simulate the whole range of an instrument. It is also necessary to interpolate one of the basic samples to each frequency requested. This may cause a noticeable change in the character of the instrument as it moves from the highest note in one range to the lowest in the next. Similarly, the strong dependence of timbre on the loudness forces us to also sample each note at many dynamic levels. Horner showed that this can be partly modelled by wavetable interpolation.[Horner 95b]

Wavetable methods are appealing because of their simple implementation. Essentially, no control information is needed, as all of the information is retained in the sample. Wavetable-based methods must be considered as a possible paradigm for analysis and resynthesis.

3.3.9 Granular synthesis and granular sampling

When the samples are very short (e.g. 20 ms) we have *granular synthesis.* This involves assembling a very large number of *grains* to form the output. This can either involve using windowed sinusoids[Xenakis, Roads 85] or windowed parts of other waveforms[Truax 87, Truax 88, Truax 90, Truax 91, Truax 93, Truax 94, Jones 88, Roads 78, Roads 85, Roads 88, Roads 91, Roads 92, Lippe 93a, Helmuth 93, de Tintis, Itagaki 96a], which is

sometimes termed *granular sampling* and is often used for time stretching.[Truax 87–94] Often the grains are specified by a higher-level process such as a stochastic approach[Xenakis, Roads 85, Roads 88, Truax 88, Helmuth 93] or cellular automata.[Miranda] Chapman discusses several other methods.[Chapman]

Granular synthesis or sampling has been implemented on the ISPW[Lippe 91, Lippe 93a, Helmuth 93], the Durham transputer network[Itagaki 96a], the NeXT computer[Helmuth 93], the IRIS MARS workstation[de Tintis], the DMX-1000[Wallraff, Truax 87–94], and dedicated DSP chips such as the Motorola 56000.[Bartoo] While designed for synthesis, *granular analysis* and *transformation* are possible. Like additive synthesis, granular synthesis requires hundreds[Roads 88], or more likely thousands, of grains to create sounds. Granular sampling has a much sparser control stream.

3.3.10 Square waves

Fourier analysis treats a square wave as a sum of sinusoids. It is equally possible to treat a sine wave as a sum of square waves. Specifically,

$$\sin(x) = sq(x) + 1/3\ sq(3x) + 1/5\ sq(5x) + 1/7\ sq(7x) + 1/11\ sq(11x) + 1/13\ sq(13x) - 1/15\ sq(15x) + 1/17\ sq(17x) + 1/19\ sq(19x) - 1/21\ sq(21x) + 1/23\ sq(23x) + 1/29\ sq(29x) + \ldots$$

where $sq(x)=sgn(\sin(x))$

Note that the coefficients do not form a regular sequence; they depend on the prime factors of the index, and may be zero. In fact, approximately 10% are zero, suggesting that making a sine from square waves is slightly easier than making a square wave from sines.[35]

The possible advantage of square waves as basis functions is that we could turn the entire wave into a large number of step functions, and then use statistical methods to allow these to evolve into the minimum set of square waves required to characterise the sound. Furthermore, the problems of round-off error and interpolation are removed with functions that are discrete in both value and time. However, their time localisation means a poor frequency localisation.

3.3.11 Walsh functions

Walsh functions are another potential set of basis functions – the transform is easier to compute than the FFT. However, like the FFT, Walsh functions are designed specifically for blocks of length 2^N. Also, as with square waves, they have poor frequency localisation.

3.3.12 Wavelets

Wavelets are short waves. Recent research has examined the decomposition of a given waveform into stretched and shifted or modulated copies of a single waveshape. Analysis and synthesis are both feasible. Wavelets appear to be worth examining, and will be discussed more in a later chapter.

---

[35] The same argument also applies to triangular waves, which, like square waves, contain all odd harmonics. The disadvantage again is that we require an infinite series of these waves to form a pure tone.

3.4 Summary

I have discussed the issue of representation, and various methods of designing a system for analysis, transcription, and resynthesis. There appear to be two possible routes that will be investigated. One is based on additive synthesis, and is the subject of chapters 5 to 7. Another possibility is through wavelets, and this will be examined in chapter 8.

4. Computing platforms

In this chapter I first discuss the requirements for music analysis and synthesis. I then examine the available hardware and software platforms. I conclude by comparing the systems' suitability for music analysis and synthesis.

4.1 Requirements

Audio is a single-dimensional entity that can be stored in analogue or digital form, on a variety of media, with a usually generous signal-to-noise ratio. The storage and reproduction are well understood and straightforward to implement, and the current state-of-the-art in digital audio satisfies most user applications. It is a bulky representation, though – a CD-ROM can hold several encyclopaedias, but only 80 minutes of music. The major technical issues to be addressed are not fidelity or generality, but bus bandwidth, disk speed, and storage capacity, and in a few years' time increased computing power may essentially have solved these problems. However, questions remain as to how composers, psychoacousticians, and musicologists can benefit most from such computational power.

It is clear that a system to analyse and resynthesise sound will have very high computational requirements. Below I discuss these in more detail.

4.1.1 Memory requirements

A common problem in DSP is the need for storage space. CD-quality[36] sound is two channels of 16-bit samples at 44.1 kHz. This means that 8 MB of memory can hold 47.6 seconds of audio. Given that we also need space for data derived from it, and for the program itself, limited memory may limit the complexity of processing.

In some parts of the analysis procedure, the memory of the PC was indeed insufficient. This was solved by using both extended memory and disk-based virtual memory, as described in chapter 6.

4.1.2 Software requirements

The calculations required are likely to include Fourier transformation, filtering, sine generation, and convolution. These are all implemented as addition, multiplication, and table lookup.

The first practical requirements of the compiler are that it produces efficient code, and permits assembly language when necessary for speed. Both are met by Borland Turbo C[Borland] for the PC, and 3L Parallel C[ThreeL] for the C40. Inmos transputers can be programmed in Occam or another version of 3L's Parallel C.

---

[36] The term 'CD-quality' is taken to mean stereo, 16-bit linear samples at a sample rate of 44100 Hz. The term is often also applied, inaccurately, to devices such as soundcards that support this format, but in many cases the lowest 2 or 3 bits may be obscured by noise.

Languages such as C are purely sequential, but if the analysis/synthesis system is to be an accurate model of the perception or production of music, it should process the information in parallel. While listening to music, we are aware of many different processes taking place at the same time. Our attention drifts away from the elements that stay more or less the same. For example, most rock music has a repetitive rhythm on the bass and snare drums that will become part of the background until it changes or stops abruptly. In trying to model this, we need a structure that is itself parallel. Parallel C seems well suited to this goal. The parallelism in software will be much greater than the actual parallelism in hardware. For example, a parallel synthesis system might have a thread running for each instrument in the ensemble, another for global reverberation, plus others for housekeeping tasks such as memory allocation, screen output, and file access. It would be beneficial for these threads to be able to schedule themselves dynamically during run-time. It is also necessary to stop each thread periodically, using `thread_deschedule()`, in order that others can be given their chance to run.

As an alternative to *procedural* languages such as C or FORTRAN, some researchers suggest using *declarative* languages such as Lisp or Prolog.[Wiggins]

Reekie examines the software architecture requirements for *real-time* parallel DSP applications, and Dannenberg looks at the difficulties of scheduling.[Reekie, Dannenberg 91] However, our transcription system has such high computational requirements that it is not possible to contemplate real-time operation, as discussed below.

4.1.3 Arithmetic issues

4.1.3.1 Integer *v* floating-point arithmetic

The standard CD format uses 16-bit linear encoding. This, and integer arithmetic in general, allows a 100.8-dB signal-to-noise ratio only where the signal has maximum amplitude. The drop in amplitude from ***fff*** to ***ppp*** is around 40 dB, which leaves the quietest passages with an SNR of 61 dB.[37] Logarithmic encoding of the sample range, i.e. floating-point arithmetic, improves the situation, but is not supported by most hardware.

4.1.3.2 Word size

The PC has an 8-bit word, and C allows us to address memory as 8-, 16-, or 32-bit integers, and 32- or 64-bit floating-point numbers. This allows us to economise on memory costs by choosing the type of the variable to reflect its likely range or accuracy. One example, assigning only 8 bits to the phase of a sinusoid, was implemented specifically to save memory.

---

[37] Arguably it depends on the *dynamic range* of the instrument, discussed in chapter 2. The clarinet has a large dynamic range of 45 dB whereas the recorder has only 10 dB.[Patterson]

The C40 has 32-bit words, and all arithmetical operations refer to the full 32 bits. This requires the use of programming tricks to fit several values into one word. This is awkward but feasible for integer arithmetic, but more troublesome for floating-point arithmetic.

With fixed wordlengths, the possibilities of overflow and underflow always exist. A more flexible but more complex solution would be to implement integer arithmetic with a variable width. For example, if we added two 10-bit arrays, we would get an array at most 11 bits wide. Then we could check to see whether the top bits are actually used, and reduce the width to the smallest width possible. It is, however, unlikely that the potential memory saving would outweigh the complexities of such an approach.

4.1.4 Disk requirements

Digital audio processing also depends on massive amounts of disk space being available. A standard 1.44-MB floppy only holds about 8 seconds of CD-quality music, and a 74-minute CD contains 780 MB of data. Long-term storage uses the drives of two Sun workstations, whereas immediate storage is handled by the hard disk of the PC.

For real-time audio, the high data rate places high demands on the disk controller and other hardware. A 66-MHz PC can only just keep up with the playback, let alone any processing of the signal. In such cases, the disk access time should be as low as possible.

The work of Nick Bailey on a parallel version of Csound running on the T800 transputers came to the tentative conclusion that the bottleneck was the disk access rather than the calculation speed.[Bailey 90, Bailey 91] Preliminary results, presented later, give support to this hypothesis. Note that the only file being accessed in the C40 experiments described later is the input file. When an output file must be written too, the overhead for disk access will approximately double. It should also be noted that all tests refer to mono sound files, implicitly halving the amount of data dealt with. The perception of stereo sound will not be dealt with until an effective treatment of monaural sound has been developed.

4.2 Platforms

There are many other platforms for computer music. Pope gives a good overview and comparison of hardware, and Gareth Loy examines the software.[Pope, Loy] General-purpose, or 'off-the-shelf' computers include the Atari[Dorfman], the Commodore Amiga[Bloch], the Apple Macintosh[Perez, Erbe, Mont-Reynaud 93], the IBM PC, UNIX workstations[BrownG 94b, Mellinger 91b, Pacheco], the NeXT machine[Helmuth, Mellinger 91b, Wang, Stainsby], the SGI Indigo[Freed 93a, Bargar 92, Rodet 92a], and the Cray supercomputer.[Kriese] Specialised systems designed for music include the DMX-1000[Wallraff, Truax 88, Pennycook 86] (with a DEC PDP-11 host), the i860-based IRCAM Signal Processing Workstation[Lindemann 90, Lindemann 91, Puckette 91a, Lippe 91, Doval, Maggi], and the IRIS MARS workstation.[Armani, de Tintis] Other platforms have been developed around DSP chips, including the Analog Devices 21020[Analog] and 21060[Vercoe 96], the

Motorola 56001[Bartoo, Nieberle, Bosi, Feiten 90] and 96002[Motorola], the Intel i860[Silberg], Inmos transputers[Bailey 90, Bailey 91, Parash, Itagaki 94, Itagaki 95a, Itagaki 96a], the C40[Nunn 94], and other chips in the Texas Instruments TMS320Cxx range.[Jones 88]

Hardware can be custom-designed for a particular task[Wawrzynek 84, Wawrzynek 91, Jansen 91], but this has the obvious disadvantage that the overall design becomes intimately dependent on the hardware. As a result, the tool can only be used by a very small number of people, and cannot be ported to another platform easily. The punched cards and state-of-the-art-then electronics that allowed the earliest computers to make music have only sentimental value today.

There were four environments available:- a TMS320C40 in a PC, a standalone PC, a transputer network hosted by a PC, and a UNIX workstation. Below I describe and compare these platforms.

4.2.1 PC

4.2.1.1 PC software

All programs used the DOS operating system. Windows 3.1 was available from the start of this work, but was not used for any of the programs developed. In Windows, allocation of system resources and control of devices is more problematic than with DOS. Windows 95 was not released until the later stages of this work, and was never used. Some of the programs developed were later found not to work under Windows 95.

Almost all the PC programs were written in Borland Turbo C.[Borland] This allows easy access to graphics, a feature not shared by the C40 and transputer platforms. The desired parallelism is not available with a sequential language such as C, but by way of consolation it may be noted that sequential programs are much easier to debug than parallel programs. Some shorter programs were written in Microsoft QBasic.[Microsoft] Other analysis was carried out using the mathematical word processor MathCad, which unfortunately cannot read large enough arrays for audio data.[Mathsoft]

4.2.1.2 PC hardware

The PC experiments were performed on one of three machines, summarised in the table below. 'Dan' is Dan Technology, and 'CMC' is Cambridge MicroComputers.

Name	*Brand*	*Processor & speed*	*RAM*	*Disk*	*Graphics*	*Other*
Wendy	CMC	486DX 50 MHz	4 MB	250 MB	non-VESA SVGA	internet access
Dan	Dan	486DX2 66 MHz	8 MB	340 MB, 1.2 GB	VESA SVGA	2 MB hardware disk cache, Gravis UltraSound, tape drive, KEE MIDI interface
Lab	CMC	486DX2 66 MHz	16 MB	340 MB	VESA SVGA	Gravis UltraSound, internet access, GUS MIDI interface

Table 14 - Specifications of PCs.

The limited memory of the PC means that some stages require virtual memory. This is described in more detail in the next chapter.

4.2.1.2.1 VGA/SVGA graphics

All of the PCs used had standard VGA graphics, which are supported by Turbo C, and various forms of SVGA graphics, which are not. SVGA (Super VGA) is a description rather than a standard, and at the lowest level each chipset requires individual support. VESA (Video Electronics Standards Association) offers the programmer a more standardised interface.

There are three ways of using SVGA graphics:-

- hand-coded low-level calls for a specific chipset
- VESA calls (using UniVesa if needed)
- an SVGA BGI (Borland Graphics Interface) library

All of these were used at one time or another. The first option was used on 'Wendy', whose Trident chipset is not VESA-compliant. This entailed finding detailed specifications for one particular chip.[Feldman] Since the C40 was to be used in both the 'Wendy' and 'Lab' PCs, the code was then unusable on the 'Lab' PC. Since two of the three PCs were VESA-compliant, I opted to use UniVesa with Wendy; UniVesa implements the VESA standard in software.[SciTech] For the standalone PC programs, the second and third options were used. The SVGABGI drivers are written by Jordan Hargraphix Software.[Jordan]

4.2.1.2.2 Mouse

Turbo C lacks mouse support but this can be implemented by DOS interrupts.[Feldman] None of the various parts of the transcription system require a mouse, but it is used as the main input device in the User Interface for the later system described in chapter 8.

After developing routines for mouse support, an interesting diversion was the construction of a 3-dimensional input device that I named a BiMouse. This is described in Appendix M.

4.2.1.2.3 Gravis UltraSound

The Gravis UltraSound (GUS) card, made by Advanced Gravis Computer Technology Ltd., was used to record and play sound.[Gravis] The original model, used in the 'Lab' machine, has a separate 16-bit recording daughterboard. It was installed in either the 'Wendy' or 'Lab' PCs. A later version, the 'Gus Max', was also used in 'Dan' or 'Wendy'. It is the same except that the recording daughterboard is built-in.

While nominally a 16-bit card, the actual resolution has been estimated at around 13/14 bits. The GUS is reckoned to be one of the best low-end soundcards. Two precautions should be taken, however; the

card should be situated away from other cards (particularly graphics cards) to minimise interference, and any DC offset should be removed.

4.2.1.2.4 Speaker

While PCs may have a variety of sound devices, they can all be relied upon to have the simple PC speaker. Although originally designed for bleeps, clever reprogramming of its timer chip allows it to play audio. It does so with an accuracy of approximately 6 bits, and has an extremely poor low-frequency response.

4.2.1.2.5 CardD

This is a high-quality A-D/D-A card, made by Digital Audio Laboratories, used occasionally in a PC in the Music Department.

4.2.2 Texas Instruments TMS320C40

4.2.2.1 Overview

The TMS320C40, normally referred to as the C40, is a Texas Instruments[38] chip designed specifically for digital signal processing.[Texas] It is a MIMD (multiple-instruction multiple-data) processor. It resides on a Transtech TDMB410 board installed in a PC.[Transtech] (A TDMB409 board was used in earlier experiments.) The architecture of the C40 is shown in Figure 31.

The raw theoretical computational power of the C40 is around 25 Mflops, but the overheads of input/output will reduce this.

[38] The addresses of all companies mentioned in this thesis are given in the references section.

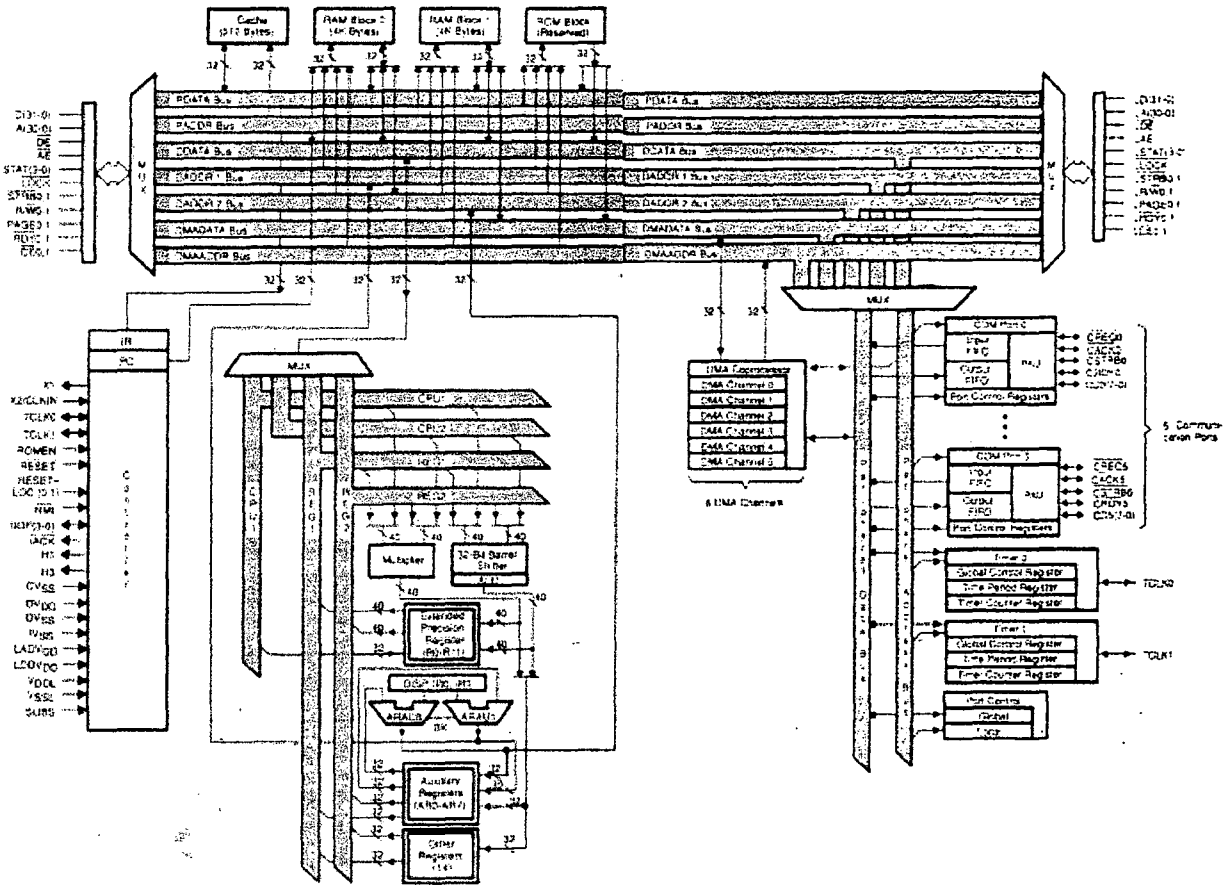


Figure 31 - C40 architecture.

4.2.2.2 Parallel C

The code is written in 3L Parallel C.[ThreeL, Culloch] Over the course of this research, the laboratory acted as a beta test site for this language, and used versions 1.0.0, 1.1.0, 1.1.1, 2.0.0, 2.0.1, and 2.0.2. Parallel C implements extensions to the TI compiler. These allow parallelism at the software level – a 'master' program actually time-slices between several tasks. Although this may not be ideal, it is acceptable for our purposes since the processor itself is sufficiently fast. The language allows *priorities* to be specified for each task – versions 1.* allowed two priorities – urgent (0) and non-urgent (1). Versions 2.* allow eight levels from 0 to 7, and also allow threads to dynamically change their own priority at run-time.

4.2.2.3 Performance and parallelism

The C40/Parallel C environment allows four flavours of parallelism.

At the lowest level, there are several instructions that carry out two operations in parallel, for example allowing two numbers to be added while a third is stored. This is taken care of by the optimising compiler.

Second, the compiler allows multiple threads within the same task. These are created dynamically at run-time (a feature exploited in the multirate spectral analysis described later).

Third, many tasks can be placed on a single processor through the use of configuration files. Both this and multiple threading require the processor to carry out "context switching" – saving all registers and loading a new set.

Fourth, several C40s can be combined to form a network of greater computational power (the improvement will be less than proportionate). Many C40s can be connected in parallel, and this is supported by message-passing over high-bandwidth links. The current implementation, however, is a single C40, and our intention is to simulate a larger network in order to examine its feasibility.

The second of these, the ability to create processes at run-time, is the most critical. Although the chip is inherently sequential, the programmer can treat it as being a flexible parallel system. This *soft parallelism* contrasts with the *hard parallelism* of other parallel computers, such as the 160-transputer network[Bailey, Itagaki 94, Itagaki 95a, Itagaki 96a] in which the hardware configuration necessarily plays a major role in defining the software setup. Instead, we can use the software to define the 'virtual hardware'.

4.2.2.4 Computational power

The high computational needs would appear to favour the C40 setup, as it is optimised with this in mind. There is also a need to provide large quantities of control information, and again the C40 is promising for its high-bandwidth interprocessor communication. It is difficult to calculate the number of operations required, as this depends entirely on the method of synthesis, but we can easily determine the maximum possible *real-time* performance as follows:-

	Processor maths performance[Texas]	25 Mflops[39]
	Sample rate	44.100 kHz
=>	Calculations per sample period	567 flops

This makes the assumption that the bottleneck is in the arithmetic calculations rather than in the passing of control information. Whether this is true or not remains to be determined.

4.2.2.5 Control of PC peripherals

Although the computational power of the C40 platform is high, the throughput is limited by the speed of the PC host and its disk and graphics subsystems. Comparing performance of the programs on several PCs showed that faster disk and graphics subsystems improved performance.

It is rather awkward to control the PC graphics from the C40, as Parallel C does not implement any graphics primitives. Therefore, routines were developed to allow the use of VGA, and later SVGA, graphics. These were based on demonstration CGA programs provided by 3L, and entailed keeping a

[39] flops = floating-point operations per second, a measure of arithmetic computational power.

complete copy of the screen in C40 memory. A background task refreshes the PC screen at regular intervals. However, this solution is inelegant, rather slow, and wasteful of the C40 memory.

Routines were also developed to allow the C40 to play sound files on the PC speaker. This was done by first loading a TSR called RESPLAY[Cox], then by calling a specific PC interrupt.

The Gravis UltraSound soundcard was described earlier. It was in the same PC as the C40 host, so code was written to allow sound to be sent from the C40 to be played on the GUS. This used code supplied by Michael Chen.[ChenM]

4.2.2.6 Custom DAC

Towards the end of this work, an output board for the C40 was designed and made by Milos Kolar in order to investigate the possibilities of real-time synthesis. The hardware is shown schematically in Figure 32.

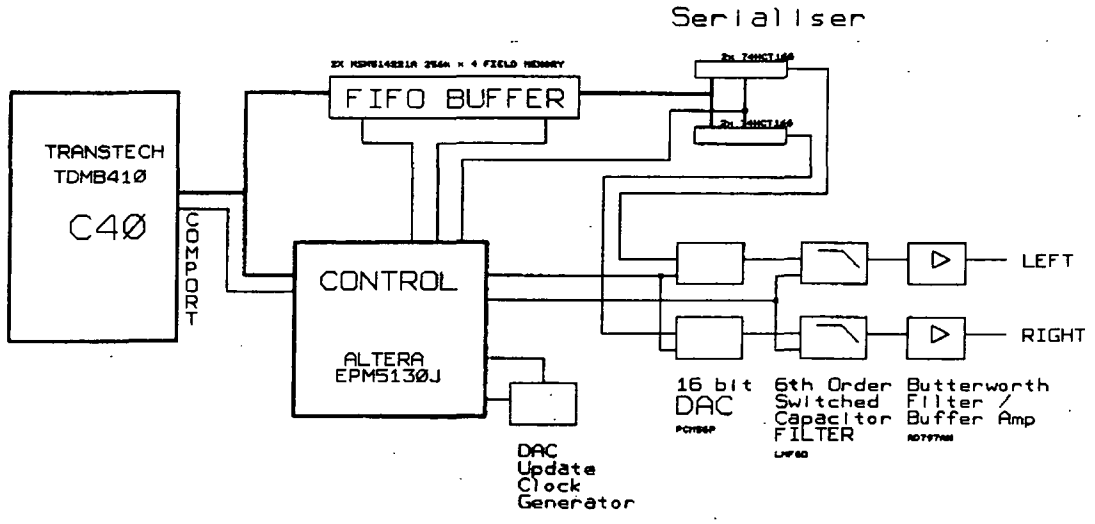


Figure 32 - C40 output system.

The board is designed around the firmware of the Altera EPMS130J, as shown in Figure 33.

The board can operate at three sample rates; 32, 44.1, and 48 kHz. The clock circuit is shown in Figure 34.

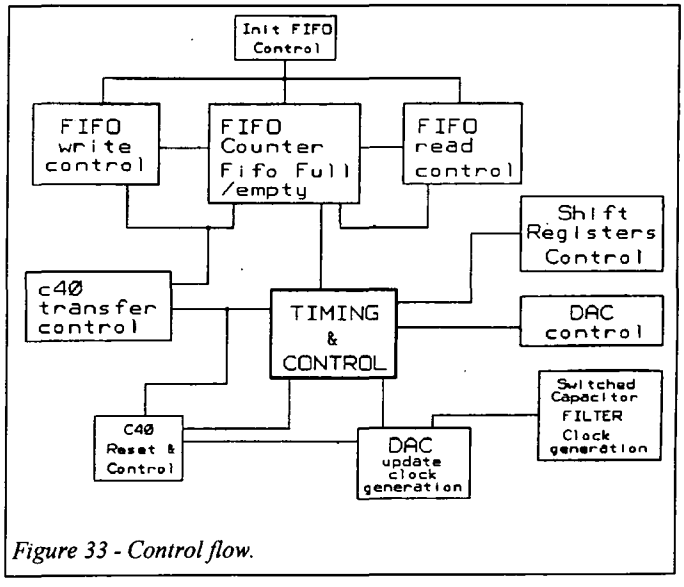


Figure 33 - Control flow.

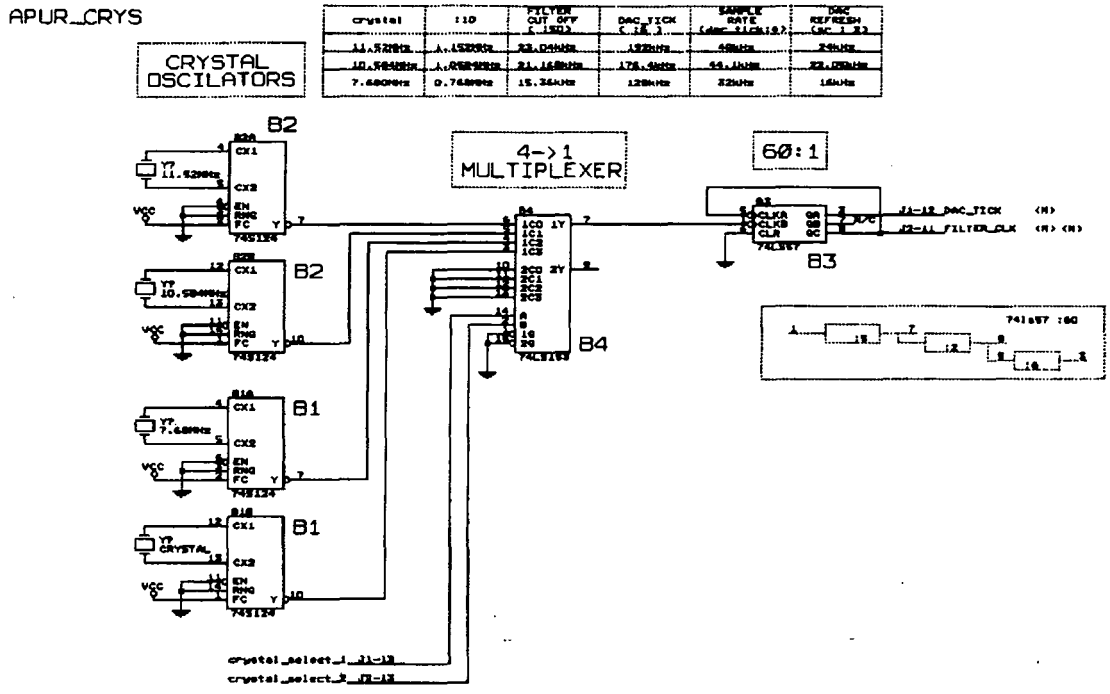


Figure 34 - Oscillators for the three sample rates.

The DAC buffering is shown in Figure 35, and the DAC subsystem is shown in Figure 36.

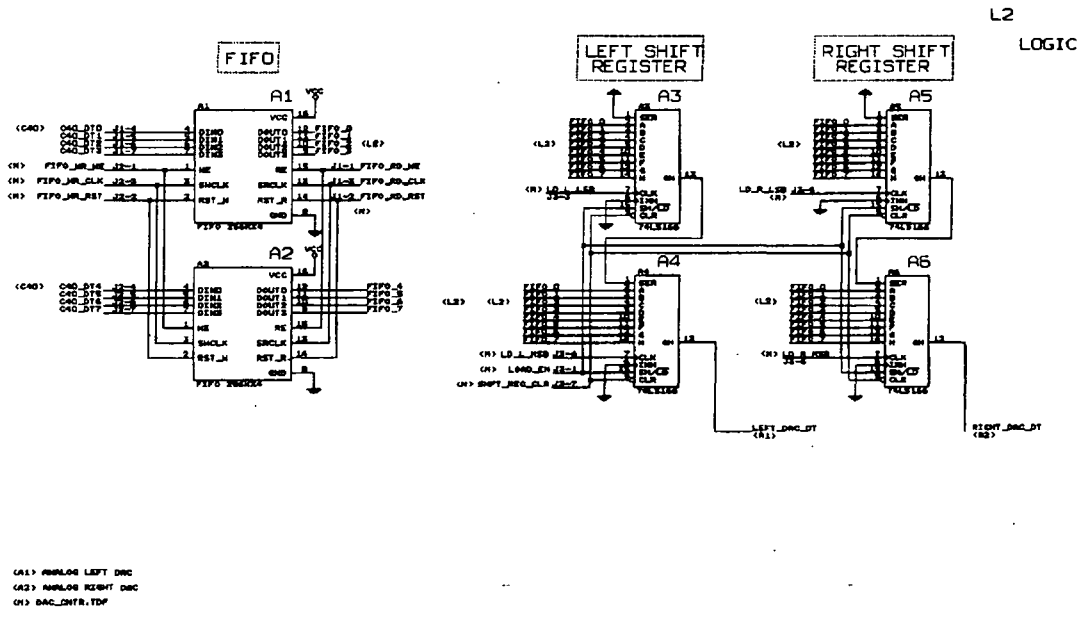


Figure 35 - DAC buffering.

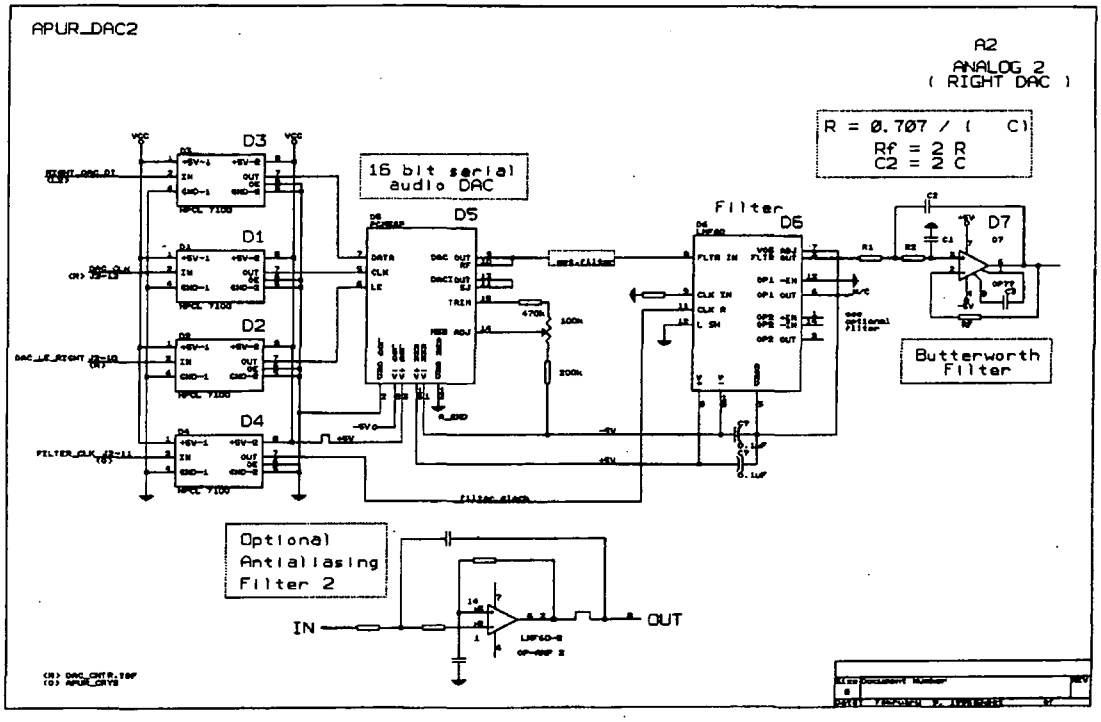


Figure 36 - DAC subsystem.

In software, the output system is addressed using the command `link_out_word(...)`. As resynthesis on the C40 had not been implemented, there was little opportunity to utilise this in the transcription system described in the next two chapters. However, it was used in the wavelet-based system described in the following chapter.

4.2.3 Transputer network

A network of 160 Inmos T800 transputers has been investigated in this research group. Nick Bailey presented a transputer implementation of Csound[Bailey 90, Bailey 91], and Takebumi Itagaki has implemented additive synthesis[Itagaki 94, Itagaki 95a] and granular sampling[Itagaki 96a] on transputers. The programs can be written in the relatively low-level language Occam, which allows little control of PC peripherals and the DOS environment[Occam], or in 3L Parallel C. Transputers have also been used elsewhere for additive synthesis.[Parash]

4.2.4 Unix workstation

The fourth alternative was to use a Sun workstation. As UNIX is a multi-tasking operating system, it allows the possibility of parallel processing. However, the Sun lacks 16-bit sound capability. (Also, this author was less familiar with UNIX programming.)

Another relatively cheap way to obtain a reasonable computational power is to use a network of Unix workstations, such as are typically available in universities.[Mellinger 91b, Pacheco, Kashino 95a] However, while this may offer high performance, the overheads of network communication, and the dependence of available processing power on the number of other users, make this approach unrealistic for real-time applications.

4.2.5 Widget technology

It is dangerous and unproductive to turn our larger task of understanding music into a search for faster, bigger, more powerful computing tools. Hence a couple of caveats are in order.

Axiom 1 - Technology increases steadily

Every computer meets one or more of the following requirements:-

- higher speed than its predecessor
- more storage than its predecessor
- more programmability than its predecessor

It also meets one or more of the following:-

- lower speed than its successor
- less storage than its successor
- less programmability than its successor

It is only recently that the power of commercially available hardware has become sufficient to handle the sheer mass of calculations required for analysis or synthesis of audio. The good news is that the level of technological development shows little sign of slowing down, and in ten years' time, it will probably seem laughable to use a machine as primitive as a 50-MHz 486DX. (At the start of this research, the highest specification available was a 100-MHz 486, and the Pentium, or 586, had not been released. At

the end, 200-MHz Pentiums are available and various 686 processors exist.) The practical viewpoint is therefore that, although there are benefits in optimising software to run 10% faster, this is only equivalent to, and much more complicated than, waiting for a processor that is 10% faster.

Axiom 2 - Technology stays the same

When the 10-GHz 100-Gbyte 986XX is released, users will still be able to complain that it's not powerful enough to handle more than twenty virtual orchestras without slowing down the raytracing of the reflections on the trombones. In other words, the more powerful a system is, the more computationally demanding the applications written for it are, and there will always be programs that run for days before giving a result. Again, there is much more to be gained in making an algorithm smarter than in making it faster.

4.3 Discussion

4.3.1 System comparison

The four platforms described above are representative of the approaches to large computing problems.

The IBM PC offers several advantages over the other setups. First, being so commonplace, it allows the use of a massive range of commercial and public-domain software. Second, it offers the potential for widespread implementation of any software produced. Third, external hardware is likely to be designed for compatibility with the PC. However, being a general-purpose computer with a general-purpose operating system, its performance in specialised applications will not be the highest.

The C40 and the transputers do not have the above advantages, but both offer much higher computation power. C40s can be connected in parallel in order to increase their power (the increase is usually less than proportionate), and they can also use time-slicing to permit greater parallelism in software than exists in hardware.

At present, only one C40 is used; this may later be extended to form a network of six to ten C40s, partially satisfying (and no doubt stimulating) the hunger for more megaflops.

Whereas the C40 network gives coarse-grained hardware parallelism, the transputer network offers fine-grained parallelism. The current setup has 160 transputers, and is ideally suited to computation that is on a massive scale but requires relatively simple programs in which the parallelism is fixed.

The theoretical computational power and other features of the four systems are as follows:-

	TMS320C40 + PC	*PC*	*Transputers + PC*	*Sun*
Mflops/processor	25	0.8 [40]	2 [41]	3.37 [42]
Mflops (total)	25	0.8	< 320	3.37
Memory	8 MB [43] (+ PC)	640 kB + 4-16 MB	640 kB [44]	24 MB
Graphics	via PC	VGA/SVGA	none	via Xwindows
Sound	Custom DAC	Speaker, Soundcard	Custom DAC	8-bit μ-law
Language	Parallel C	C	Occam/Parallel C	C
Parallelism	Hardware/Software	none	Hardware	UNIX OS

Table 15 - Comparison of the four computing environments.

The task we are undertaking – the analysis, transcription, transformation, and resynthesis of polyphonic sound – is a computationally large task. It requires a high processing power, abundant memory, and support for graphics and sound. None of the available systems meets all of the requirements fully.

4.3.2 Operation in time

We have insufficient processor power and memory for real-time operation. Each minute of music may require many hours of computation. Below I examine how processing takes place with respect to the time axis.

4.3.2.1 Real Real Time

This class contains processes that can be solved within an arbitrarily small time interval after the input has been presented. An example is a program that converts text to Morse code. We can guarantee to create the output after an arbitrarily small delay, given a sufficiently fast processor.

4.3.2.2 Delayed Real Time

A further class of procedures are those that are guaranteed to produce results after a finite and known delay. A typical case is where we use the STFT to transform data from the time domain to the frequency domain. Here, we cannot determine the frequency content of the wave from T_1 to T_2 until after T_2. In other words, we can never determine "there is a 123 Hz wave now"; we can only determine "there was a 123 Hz wave a moment ago" (and the size of the moment increases for lower notes). Another example

---

[40] This value was determined by writing a short C program to time two loops, one consisting of fetch/write instructions on (single-precision) floating-point numbers, and the other consisting of fetch/multiply/write instructions. It was assumed that no inefficiency was added by the C compiler.

[41] These figures are for a 20-MHz processor. As noted earlier, the performance of N processors is usually less than N times the performance of one. Thus the figure is certainly an overestimate.

[42] This figure is derived using the Linpack benchmark. [Athena]

[43] For comparison, this figure is for an 8-bit byte. The C40 in fact has 2 megawords of 32-bit words.

[44] This is the total. Each transputer only has 4 kB of memory.

is filtering - since the filter must be causal, there will inevitably be a fixed delay. Such processes can never run in real real time, regardless of the processor power.

4.3.2.3 Slow Real Time

In this category are processes where we have a finite but insufficient processor power. When the processing cannot keep up with the input data, it delays reading the input until the output has been calculated. However, the process still operates in monotically increasing time. Such processes can be made real real time or delayed real time by increasing the processing power.

4.3.2.4 Unreal Time

This category includes 'off-line' processes that do not operate with respect to a monotonically increasing time. For example, both perfect low-pass filtering and perfect Hilbert transformation require an infinitely long non-causal filter response, so there is a theoretically infinite delay from output to input, as we require all past *and future* values to calculate the output. Even on an arbitrarily fast processor, real-time performance is impossible. As Jaffe puts it, you cannot make something laugh before you tickle it.[Jaffe]

Another example, discussed later in more detail, is in transcription. Most notes have a rapidly-changing attack followed by a steady state and a decay portion. I would hypothesise that reversing the entire waveform would make it easier to detect the attacks, as we could determine the frequencies during the reversed decay and wait for them to disappear abruptly at the reversed attack. This technique is also suggested by Serra, who tracks audio by analysing reversed blocks.[Serra 90] However, even if this were shown to give accurate transcription results, our own perception is in normal, continuously increasing, time. Thus, this cannot be said to be a valid model of our perception.[45] Another drawback in this case is that we can never use such methods for continuous input, or for a data stream of unknown length.

4.3.2.5 Discussion

One way to proceed is to use 'slow real time', analysing the music once very slowly. The second way is in 'unreal time', allowing many passes through the data, with each pass building on the knowledge from the previous pass. The first option is preferable.

4.3.3 Summary

I have outlined the specifications and features of the platforms available, and discussed design options for a non-real-time system. The transcription system will as far as possible be designed as a 'slow real time' system, in that the outputs are calculated in the same order as the input is presented. However, a few parts are in the 'unreal time' category, such as compensating for the filter lags by sorting. In total,

---

[45] However, it has some similarities with the process of manual transcription. Indeed, a human transcriber will replay short segments of a piece, many times, concentrating on whatever was not transcribed on the previous pass, as it is impossible to even write music in CPN in real time.

the processes from raw audio to MIDI takes around 800 times real time. Considerable optimisation could still be done, and the PC routines would run faster on the C40, but it is unlikely that we will reach real-time operation.

5. Previous research on transcription and source separation

This chapter presents an overview of previous research on transcription and source separation. I first briefly outline the applications for such systems. I next review monophonic and polyphonic systems, and other closely-related research. Finally I compare the approaches.

5.1 Motivation and applications

The ability to convert a piece of polyphonic music into a score-like representation is something of a holy grail in computer music research. There are several applications for this. First, it would allow musicologists to study music that has never been notated, such as improvised jazz[Hidaka], folk and ethnic music[Tsujimoto], classical and contemporary music, speech prosody, and songs of birds and whales. It would also permit the "Intelligent Editor of Digital Audio" outlined by Chafe and Foster, in which the composer can deal with phrases, bars, and notes whilst remaining in the digital domain.[Chafe 82, Foster]

Second, even when the score is known, it is useful to be able to determine exactly how the score was interpreted. Specifically one wishes to extract the precise timing, and sometimes the amplitude, of each note.[46] This is done either for expressive performance analysis[Repp, Scheirer 95a] outwith real time, or for *auto-accompaniment* systems[Vercoe 84, Dannenberg 84] in real time. Related to this is a more general *artificial performer*, where the computer interacts with multiple players. This is currently only possible with MIDI[Pennycook 93] or multiple monophonic systems.[Grubb]

Both of these are lossy processes as timbres would not be recorded – 'transcription' only describes one part of the problem. An example discussed earlier and analysed later is the didgeridoo[47], which sustains a single note[48] for a long time. Clearly we could transcribe this into a single MIDI or CPN note, and equally clearly this would omit almost all of the important information.

The third application entails encoding the timbres of each note, accurately or (in many cases) approximately. This provides a tool for analysis and *re*synthesis, or *synthesis-by-analysis*, and the intermediate format would allow musically interesting and useful transformations, in conjunction with

---

[46] Another application of expressive performance analysis is to music videos. Often the singer on a rock video is poorly synchronised with the music. To solve this, we would need to analyse two audio files - the actual music played (or, preferably, the vocal track alone), and the music as sung during the recording of the video. It would then be possible to match these two in order to determine when the singer was rushing or dragging, and to drop or repeat video frames to achieve audio-visual synchronisation to within 20 ms, which is unlikely to be noticed. Typical quoted figures for TV are 100-150 ms, although this depends on the context; tennis coverage, for example, would require a lower figure. (A curious side-result is that the two performers need not be the same person, allowing anyone to mime perfectly to someone else singing.) Similar applications exist in other multimedia fields, especially teleconferencing, where video and audio are transmitted separately, but must be presented together.

[47] Alternatively spelled didjeridu.

[48] The third and fifth harmonics can also be played with some difficulty, but this is not normal playing practice.

the 'intelligent editor' mentioned above. Related applications include the removal of clicks and noise[BergerJ 94b], where we are not concerned with the separation of musical sources, merely the separation of 'all the music' from 'all the noise'.

Fourth, there is a desire to objectively examine aspects of our own perception of music. Whilst single-note phenomena are generally understood, our innate understanding of music as a large structure of notes and timbres has only recently received attention. As most music is note-based, the ability to examine perception of chords, melodies, rhythms, and other structural entities depends on first being able to understand notes. Scheirer observes, "there is an implicit assumption that the musical score is a good approximation to the mid-level representation for cognitive processing of music in the brain".[Scheirer] Thus, we are in a sense trying to model the way we perceive sound. Some researchers[Leman, BrownG 94a] use physiologically accurate models of the human auditory system. However, in this work I would argue that implementing the imperfections of the human auditory system on a computer model may introduce unnecessary complications. The essential part of the problem would remain unsolved – how does one decompose a single auditory stream into multiple parallel streams?

5.2 Monophonic transcription systems

When the signal is monophonic, the problem, while not trivial, is certainly much simpler, and is often termed *pitch detection, pitch extraction,* or *fundamental frequency estimation.* This is done to develop pitch-to-MIDI systems, usually in real time. Here the aim is to estimate the pitch and amplitude with as small a latency as possible. There is a wide range of techniques used, including the cepstrum method[Noll, Pabon 94b], autocorrelation[Sondhi, BrownJ 91b], harmonic methods[Schroeder, Amuedo, Piszczalski 77, Piszczalski 79, Piszczalski 81, Terhardt 79, Terhardt 82a, Todoroff, Hermes, Fernández-Cid], the average magnitude difference function[Ross], fundamental period measurement[Kuhn], tunable digital filters[Lane], the modal distribution[Sterian], linear prediction[Maksym, Rabiner, Markel], least-squares[Choi], neural networks[Taylor 94, Taylor 95, Sano], and wavelets.[Kadambe] In some cases, the timbre of the instrument is known in advance, or it is assumed to be harmonic. Often the output is quantised to the twelve-tone scale, making it unsuitable for continuous-pitch instruments such as the trombone and the human voice.

The 1984 ICMC saw two monophonic systems applied to *real-time auto-accompaniment,* in which the score is usually also known in advance. Roger Dannenberg's system was developed at the Massachusetts Institute of Technology[Dannenberg 84, Block, Dannenberg 88, Dannenberg 91] and was used with trumpet and flute. Lorin Grubb developed a related system[Grubb] that allows a computer to play as part of an ensemble. It tracks up to four acoustic instruments (monophonically) in order to estimate the 'average' score position. Barry Vercoe, also at MIT, developed several systems using the 4X computer for tracking solo instruments. His first system[Vercoe 84] was designed for auto-accompaniment of a solo flute. With such systems the score is known in advance and the computer accompanist knows what details are supposed to be in the input sound. It used not only the acoustic information but also optical

sensing of the keys in order to derive pitch estimates. He notes that the pitch estimate depends on the presence of the fundamental – as a result, tracking of multiphonics[49] is impossible. Vercoe and Miller Puckette later developed this into a system[Vercoe 85] for auto-accompaniment of solo violin. It was also able to learn from *rehearsal* with the soloist. Again, however, it worked well for purely monophonic sounds, but could not deal with double stops. Wake at Osaka has examined auto-accompaniment in the context of a real-time jazz system.[Wake 92, Wake 94]

Analysis of song allows previous work on speech recognition to be reapplied. Research at Osaka University has examined transcription of a Japanese folk song.[Niihara] This uses *a priori* knowledge of the particular Japanese scale used. At Waseda, Inoue developed a *karaoke* system that allows the singer to modify the tempo.[Inoue 93, Inoue 94] It is based on speech recognition, and uses both score information and lyrics data. It can also correct the singer's pitch errors. Auto-accompaniment systems have also been developed by Naoi at Waseda and Horiuchi at the Tokyo Institute of Technology.[Naoi, Horiuchi 92, Horiuchi 93] Instead of precisely matching the tempo fluctuations of the performer, Horiuchi's system has an *independence* parameter that varies dynamically from 0 (where it follows the performer exactly) to 100 (where it plays by its own rules). Peter Pabon also developed a real-time system for analysis and resynthesis of the human voice.[Pabon 94a]

Judith Brown at MIT developed a transcription system using "narrowed" autocorrelation.[BrownJ 87, BrownJ 89, BrownJ 91a, BrownJ 91b, BrownJ 92] She shows that narrowed autocorrelation, which incorporates higher harmonics into the pitch tracking, is more accurate than conventional autocorrelation.

Xavier Serra reports on a sound transformation system using a 'deterministic plus stochastic' decomposition.[Serra 89, Serra 90, Serra 96] While timbres (both harmonic and inharmonic) are modelled by conventional spectral methods, noise components are modelled by filtered noise – the stochastic part – such that phase information is deemed unnecessary.

Perry Cook describes a transcription system[Cook 92, Cook 93] for *real-time* use on valved brass instruments. It makes assumptions about the playing range, and is illustrated for the trumpet. It uses a Period Predictor Pitch Tracker[Cook 91], another autocorrelation-based method. Like Vercoe, it uses optical sensing – in this case four-position optical sensors on each valve.

As the monophonic field is now relatively mature, there are now several commercial products that carry out pitch-to-MIDI conversion.[Wildcat, Hohner, Emagic, Opcode, AudioWorks]

---

[49] Multiphonics are two or more notes played (often with great difficulty) on a conventionally monophonic instrument. Flutes, saxophones, horns, and trombones are capable of playing multiphonics.

5.3 Polyphonic transcription systems

We now turn to polyphonic transcription, a task acknowledged to be harder because of the overlapping and interfering of harmonics from different sources. As with the monophonic case, much depends on how much *a priori* knowledge is given.

All of the systems below are monaural, corresponding to our ability to separate monaural sound – none attempts to utilise, or derive, spatial information. One might reasonably ask why potentially useful information is not used (except in the multiple-microphone systems described later); source separation of real stereo sounds is probably easier than in mono. Yet the assumption that the music was made in a real acoustic environment is not necessarily true; there is no bar to an electroacoustic composer presenting physically impossible situations such as different frequencies in the left and right ears, placing partials of a note at different spatial locations, or giving partials different vibratos.

5.3.1 Tom Stockham, MIT, 1975

Tom Stockham at MIT, and later at Soundstream, is regarded as one of the founders of digital audio. He developed a system for denoising and source separation.[Stockham 75] This was used to make the earliest digital recordings from very early analogue recordings. He used homomorphic deconvolution to remove noise and the voice of Enrico Caruso from orchestral accompaniment.

5.3.2 Tom Parsons, New York, 1976

Tom Parsons at the Polytechnic Institute of New York developed a system for separating two voices. He uses an STFT-based scheme.[Parsons] First, one fundamental frequency is estimated, and the spectral peaks corresponding to it are removed. This is then repeated for the second speaker.

5.3.3 James Moorer, CCRMA, 1975-77

Moorer at CCRMA[50] was one of the first to develop a polyphonic system.[Moorer 75, Moorer 77b] He examined the transcription of two lines played on a guitar. The two pitches were not allowed to cross, and could not be a fifth or an octave apart. His method uses heuristic processing of the outputs of a large filterbank.

5.3.4 Chris Chafe, CCRMA, 1982-86

Chris Chafe first examined the path from note list to printed score, such as recognition of the key, tempo, and time signature[Chafe 82], in conjunction with Scott Foster's work on preliminary audio segmentation.[Foster] Later, Chafe developed a polyphonic system that uses a Bounded-Q FT in a polyphonic system for piano transcription.[Chafe 85, Chafe 86] It was also applied to transformation, and for source separation of a piano and a harpsichord.

---

[50] Center for Computer Research in Music and Acoustics at Stanford University.

5.3.5 Andrew Schloss, CCRMA, 1985

Andrew Schloss developed a system for transcription of percussion.[Schloss] (The case of transcribing percussion from percussion *plus* other sounds is called 'beat-tracking' and is discussed later.) Schloss segments the data from two conga drums using amplitude thresholding, then analyses the type of stroke. Next, it parses the rhythms to determine the tempo, and produces a transcription.

5.3.6 Mitch Weintraub, CCRMA, 1985

Mitch Weintraub's system is for monaural sound separation, than recognition, of two voices, and works by grouping autocorrelation peaks.[Weintraub] This is an example of source separation *without* subsequent transcription. A common application in telephony is cancellation of interfering speech.

5.3.7 Bernard Mont-Reynaud, CCRMA, 1985-93

Bernard Mont-Reynaud describes another polyphonic system, running on the Macintosh platform with a DSP board.[Mont-Reynaud 85, Mont-Reynaud 88, Mont-Reynaud 90] The initial analysis, like Chafe, is by the Bounded-Q Frequency Transform (BQFT), a multirate approach in which Q varies within a ratio of 2. Image convolution and other image processing techniques are used to extract the pitches of a two-voice piano sonata. (A different system used frequency co-modulation to separate sources.[Mont-Reynaud 89]) Mont-Reynaud later developed this into the SeeMusic system, which includes visualisation and achieves better-than-Heisenberg resolution through the use of a *multiresolution* approach, which combines several BQFTs with different Q ranges.[Mont-Reynaud 93]

Mellinger, also at CCRMA, took a similar approach in developing SoundExplorer, an NeXT 'interactive workbench' for source separation.[Mellinger 91a, Mellinger 91b] As the process can take up to 10 hours per second of input on a single machine, he spreads the computational load over a network of workstations.

Several of the above CCRMA systems are also described in papers by Chowning.[Chowning 84, Chowning 86b]

5.3.8 Robert McAulay, MIT, 1986

McAulay and Quatieri developed a system for analysing speech.[McAulay, Quatieri 85, Quatieri 90] Their system is notable as several other speech and music systems were based upon it.[Horner 95, Depalle 93a, Depalle 93b] MQ analysis, as their technique came to be known, initially uses the STFT, but then models sound as a number of sinusoidal components. This was applied to suppression of interfering speech. Julius Smith used a similar technique in PARSHL, demonstrating its applicability to non-harmonic sounds.[SmithJ 87, Serra 90] Kelly Fitz's LEMUR system is also based on MQ analysis.[Fitz, Bargar 95] Whilst originally used for monophonic sounds, MQ analysis has also been demonstrated to be useful for polyphonic analysis.[Maher 89, Maher 90, Stainsby, Scallan] However, being a lossy process it is not well suited to exact resynthesis.

5.3.9 Haruhiro Katayose, Osaka, 1988-90

A polyphonic transcription system was developed by Katayose et al.[Katayose 88, Katayose 89] for transcription of piano, guitar, or shamisen.[51] It also uses the above peak frequency estimator. It assumes the music has a regular rhythm, without dramatic tempo deviations. It uses *a priori* knowledge of the timbre to aid in separating polyphony. The original system cannot separate *different* instruments, but a later version[Katayose 90a] reportedly separates two instruments chosen from piano, guitar, clarinet, and violin. This was developed into the Virtual Performer system.[Katayose 93, Takeuchi] Also developed at Osaka was a tool for transcription of polyphonic piano music for expressive performance analysis.[Takami, Katayose 90b] This system makes use of score information. Earlier research at Osaka produced an interactive tool for examining Southern Pacific ethnic music.[Tsujimoto] This uses a novel peak frequency estimation method using the magnitude of the inverse spectrum.

5.3.10 Richard Kronland-Martinet, Marseilles, 1987-88

Kronland-Martinet at LMA[52] at Marseilles used wavelet transforms in his sound analysis system.[Kronland-Martinet 87, Kronland-Martinet 88] Wavelets, discussed more in a later chapter, allow a constant Q. As well as sonogram-type analysis, he illustrates wavelets designed for octave detection. He notes the usefulness of wavelet analysis for timescale modification. Guillemain also uses wavelets for analysis and resynthesis.[Guillemain]

5.3.11 Andranick Tanguiane, ACROE-LIFIA, 1987-95

Andranick Tanguiane at the USSR Academy of Sciences, and later at ACROE-LIFIA[53], examined the recognition of chords.[Tanguiane 87, Tanguiane 88] He proves that the optimal factorisation of the log-spectrum of a chord is the log-spectrum of the timbre convolved with the log-spectrum of the pitches in the chord.[Tanguiane 93b, Tanguiane 93a] He also examines rhythm recognition, and again approaches the task by choosing the data representation that minimises its complexity.[Tanguiane 91] This was demonstrated for monophonic key-tapping of the snare drum part of Ravel's Bolero. A later paper combines both of these ideas[Tanguiane 96], postulating that for audio perception, the following axioms hold:-

1) The frequency axis is logarithmically scaled.
2) The ear is insensitive to phase.
3) Data can be grouped with respect to structural identity.

---

[51] The shamisen is a Japanese instrument similar to a lute.

[52] Laboratoire de Mécanique et d'Acoustique - Mechanics and Acoustics Laboratory.

[53] ACROE stands for Association pour la Création et la Recherche sur les Outils d'Expression (Association for Creation and Research on Tools for Expression, part of the Ministère de la Culture et da la Francophonie (Ministry of Culture and Language). LIFIA is the Laboratoire d'Informatique Fondamentale et d'Intelligence Artificielle (Pure Informatics and Artificial Intelligence Laboratory) at IMAG, the Institut d'Informatique et de Mathématiques Appliquées de Grenoble (Institute of Informatics and Applied Mathematics).

4) Data are represented in the least complex way in the sense of Kolmogorov (least memory storage required).

This technique of data minimisation is a powerful one – understanding a signal is largely equivalent to compressing it. Tanguiane's approach is mathematical; he does not present an implementation.

5.3.12 Barry Vercoe, MIT, 1988

Barry Vercoe and David Cumming describe a method for tracking polyphonic audio.[Vercoe 88] It uses the *Connection Machine* (CM), a highly parallel SIMD computer with 65536 processors, and audio algorithms are carried out using an extension of the Universal Processing Element (UPE) of Carver Mead.[Wawrzynek 84] It first carries out the Short-Time Fourier Transform (STFT) and also produces the rate of amplitude change for each bin. These are then grouped into separate sources by noting similarities in onset time and in small fluctuations. Processing is then carried out using an artificial neural network. However, no results are given in this early paper. He concludes that the heavy use of pattern matching is only suitable for massively parallel processing resources.

5.3.13 Bob Maher, Illinois, 1989-90

Bob Maher also examined signal separation.[Maher 89, Maher 90] He first compares the differences between speech-based and music-based separation systems, and then discusses the generalisation of the problem, observing that 'projects of this sort can rapidly fall into the trap of ad hoc, special-purpose techniques to solve a particular problem, only to find another problem created'. His system also makes several assumptions:- the polyphony is two, the timbres are nearly harmonic, the 'lower' voice may not go higher than the 'upper' voice, and the minimum note duration is known.

His approach starts by using MQ analysis. It then estimates the two fundamental frequencies. He uses four test pieces – two synthesised and two real, and while his system generally gives a good transcription, he notes that it is more susceptible to errors when partials overlap and at onsets and offsets.

5.3.14 Al Bregman, McGill, 1989-96

Bregman defines the term *Computational Auditory Scene Analysis (CASA)*, which is adopted by much of the later work.[Bregman 89, Bregman 96a, Bregman 96b] In particular he examines the factors influencing the fusion of partials, and lists these as onset and offset synchrony, frequency separation, regularity of spectral spacing, binaural frequency matches, harmonic relations, parallel amplitude modulation, and parallel gliding of components. These factors contribute to the *primitive grouping* of partials. In contrast, *schema-driven grouping* depends on our learned knowledge of instrumental sounds and musical practice.

5.3.15 Edward Pearson, Warwick, 1990-91

Like Mont-Reynaud, Pearson uses a multiresolution system[Pearson 90, Pearson 91] to detect features in polyphonic audio. The Multiresolution Fourier Transform[Calway, Wilson 92a, Wilson 92b] (MFT) is an over-complete set of windowed Fourier transforms of size 2^N for *several* values of N. Pearson uses a sample rate of 24 kHz and (at least) eleven FFTs, up to an FFT size of 2048. This allows all reasonable trade-offs between time resolution and frequency resolution to be investigated. He then applies differencing to the transform coefficients and applies a filter to enhance the onsets. Next, peak detection is used to form a set of candidate onsets. Then, the onsets over all scales are compared in order to remove spurious partials. Finally, onsets are chosen by combining the high frequency resolution of lower levels (i.e. with longer FFTs) with the high time resolution of higher levels, and the resultant partials are grouped to form notes. The technique works well on his simple example, the notes F#4 and C4 played on a piano[54], and in later work it is applied to the woodwind trio of Bach's 1st Brandenburg Concerto.[Pearson 91, Wilson 92b]

Tim Shuttleworth, also at Warwick, reports on preliminary work on polyphonic transcription of Schubert's "Trout" piano quintet and an unnamed piece by Tchaikovsky.[Shuttleworth] He also uses the MFT, but asserts that only three or four MFT scales are needed. He pinpoints a key disadvantage of the MFT, its very high data rate. The amount of original data is multiplied by the number of MFT levels.

5.3.16 Boris Doval, Paris, 1991

Doval's system on the IRCAM musical workstation uses harmonic methods. Although originally designed for monophonic sounds, it forms multiple hypotheses for the fundamental frequency, and can detect several fundamentals. He illustrates this for two clarinets.[Doval]

5.3.17 Martin Cooke, Sheffield, 1991-96

Martin Cooke at Sheffield developed a separation system.[Cooke 91, Cooke 93b] He illustrates with the separation of speech from various 'noise' sources – a pure tone, white noise, impulses, laboratory noise, rock music, a siren, a telephone, female speech, and male speech.[Cooke 93a]

Cooke questions the *bottom-up*, or *data-driven* approach of his and other earlier separation schemes, where the data flows from low-level to higher-level entities.[Cooke 96a] He argues for speech 'schemas' to be more tightly integrated into the CASA process – i.e. for the process to incorporate *top-down* processing. This question is also discussed in detail by Slaney.[Slaney 95b, Slaney 95a] Cooke also lists many of the circumstances where we can perceive speech distorted in various ways (see [Cooke 96b] for a system to analyse 'occluded' speech) but conventional (bottom-up) recognisers cannot, and suggests alternative

---

[54] Presumably these pitches are chosen a tritone apart so that their lower harmonics do not overlap. The first potential overlap is between the fifth harmonic of F#4 (5×370.0=1850Hz) and the seventh harmonic of C4 (7×261.6=1831 Hz).

architectures. It should be noted that while speech perception has some similarities with music perception, not all speech schemas are applicable to music.

5.3.18 Guy Brown, Sheffield, 1992-96

Much of Cooke's work is in collaboration with Guy Brown, and vice versa. Brown's original system was designed for separating speech from the other sources described above.[BrownG 92a, BrownG 94b] It is illustrated in Figure 37. This was later extended to analysis of synthesised musical sounds.[BrownG 94a]

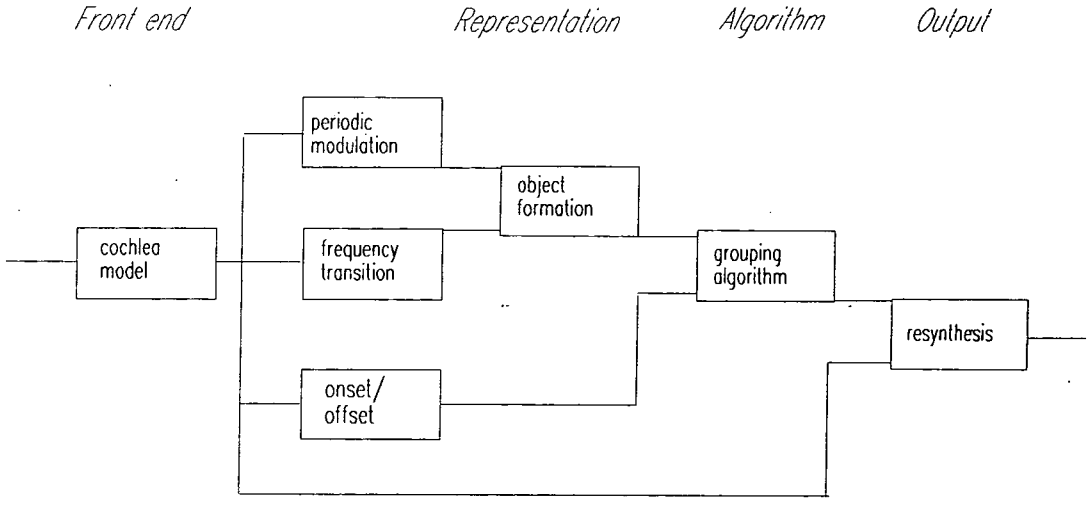


Figure 37 - Overview of Guy Brown's analysis system.

Brown first analyses the signal using a highly accurate model of the human auditory periphery. Spikes on the simulated auditory nerve are converted into five 'feature maps' representing offsets, onsets, frequency transitions, rates (essentially a spectrogram), and autocorrelation. Next, 'auditory elements', representing the amplitude and frequency movement of each partial, are extracted. Then, these elements are grouped into notes. Earlier work[BrownG 92a] used offsets and onsets; later work[BrownG 92b] uses pitch contours.

While many transcription systems stop at the stage of separating all the notes, Brown examines how all the notes *from a particular instrument* can then be grouped together, by examining the Wessel sequence[Wessel 79] played on clarinet and brass timbres. (This does not contain simultaneous notes.) He successfully groups the resultant notes according to their brightness and onset asynchrony. He then examines the same timbres played as a duet, and while the separation results are less ideal, they appear promising. It transcribes 10 out of 17 notes. Brown's approach is bottom-up, but like Cooke he suggests that top-down processing is also important, and this is developed in Crawford's 'interactive' system.[Crawford] Later work turns to a neural oscillator model.[BrownG 95a, BrownG 95b, BrownG 96]

5.3.19 Dan Ellis, MIT, 1991-96

Dan Ellis of MIT, and later the International Computer Science Institute, describes a system for characterisation of sounds.[Ellis 91, Ellis 92a, Ellis 92b, Ellis 93, Ellis 94, Ellis 95a, Ellis 95b, Ellis 96a, Ellis 96b]

He discusses several previous *data-driven*, or *bottom-up*, models[Weintraub, Cooke 91, BrownG 92a], and argues that this approach cannot deal with auditory illusions where 'the perceived content of the sound is in some sense incorrect or different from what was actually presented'. An example is the *continuity illusion,* described in the previous chapter.[Bregman 89] Ellis uses the continuity illusion to back up his case for *top-down* processing, where higher-level information is used in the interpretation of lower-level data. (see also [Bregman 96a, Bregman 96b])

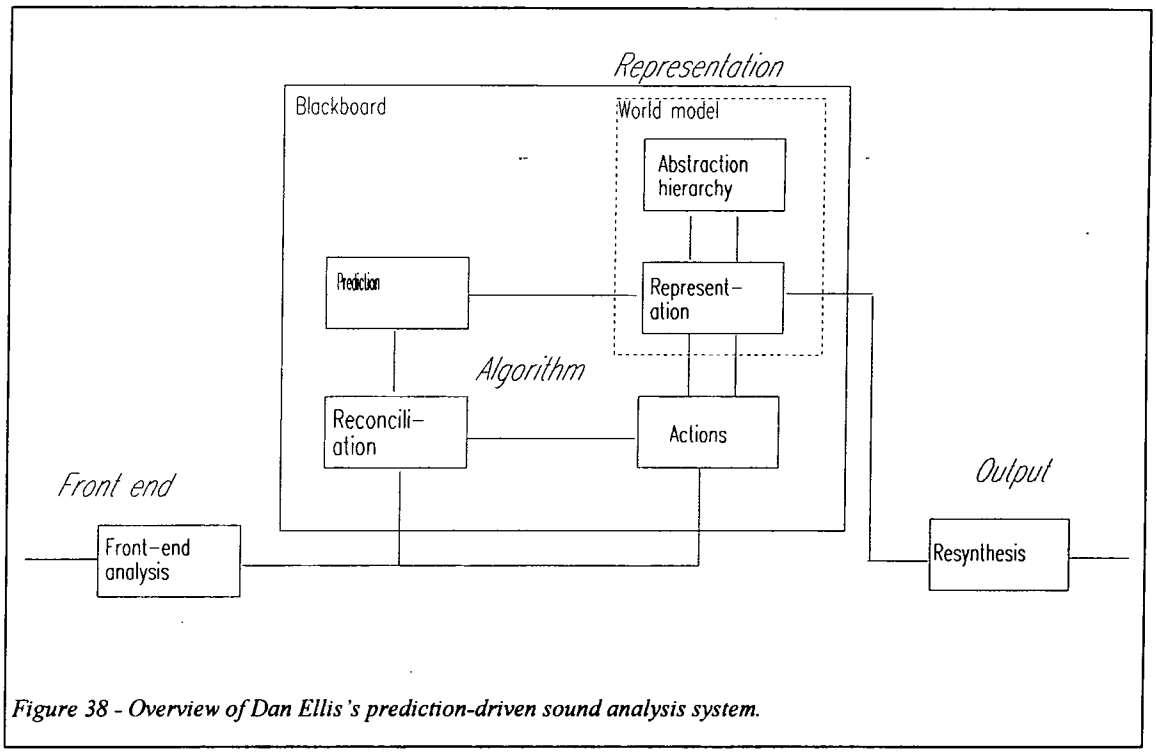


Figure 38 - Overview of Dan Ellis's prediction-driven sound analysis system.

His system is shown in Figure 38. Compare it to Guy Brown's 1992 system shown above.

In his system, the front-end analysis provides several outputs:- an overall signal intensity, a set of filterbank outputs, the autocorrelation, an onset map, and the correlogram (the autocorrelation within each frequency channel).

He uses three sound elements to represent the sound – noise clouds with separate temporal and spectral contours, transient clicks, and *wefts* representing wideband periodic energy.[Ellis 96c] The reconciliation engine uses a blackboard system based on IPUS[Carver] and changes the information representations according to the match between it and the front-end output.

He applies his system to a male speaker in a noisy environment (which forms two wefts for the vowels in "bad dog", three clicks for the consonants, and a noise background) and 'construction-site ambience'

with noises, clicks, and bangs (which display the system's ability to characterise arbitrary and dense sound scenes). Resynthesis, he reports, is more problematic, and the reconstructed sounds are still distinguishable from the original.

5.3.20 Kunio Kashino, Tokyo, 1992-93

Kunio Kashino describes a monaural sound source separation system.[Kashino 92, Kashino 93] He makes the useful distinction between a *physical* sound source and a *perceptual* sound source. A piano note has three physical sources (each string) and one perceptual source, whereas recorded polyphony has one physical source (the loudspeaker) but many perceptual sources. I have discussed this elsewhere in relation to the sound of a violin section.

He also discusses and dismisses methods based on spectral 'templates' as this requires a complete set of all musical sounds that *might* be present. Such a collection could never be completed as new (especially electronic) instruments are being developed daily. As a result of these considerations, their system assumes a monaural signal, and no *a priori* knowledge.

The initial frequency analysis uses IIR filters (which cannot allow perfect reconstruction), at logarithmically spaces frequencies. It then uses a pinching plane method to track the times and frequencies, and clusters the onsets according to the degree of harmonic mistuning and onset asynchrony.

They develop their system to examine three cases: one where no timbral information is known or used, one where timbral information is derived during processing, and one where timbral information is known in advance. The test signals are 2- or 3-note chords consisting of one flute note and one or two piano notes, played by wavetable synthesis. Their system appears to be able to correctly identify most of the chords.

Kashino's later work[Kashino 95a, Kashino 95b] also adopts an approach that combines bottom-up and top-down processing using Bayesian probability networks. Higher knowledge includes statistical analysis of chord transitions and joint chord-note probabilities, and instrument spectra. The grouping employs common harmonicity and common onsets, and separates two- and three-note mixtures of flute, clarinet, trumpet, violin, and piano with a 'recognition rate' of 50-90%. It is implemented on a combination of a Fujitsu AP1000 parallel computer and a network of other workstations.

5.3.21 Jeff Pressing, La Trobe, 1993

Jeff Pressing at La Trobe University in Melbourne developed a transcription system for the Macintosh.[Pressing 93a] It is an *interactive* system, in that 'the user selects the musical and cognitive contexts appropriate to the interpretation of his or her actions'. This means that the user specifies the barlines, the time signature, and so on. These are connected to the onsets detected in the audio.

Information that is not specified by the user can be derived from the audio, although few details of this procedure are given.

5.3.22 Alain de Cheveigné, Paris 7, 1993-96

Much research in speech has gone into the separation of concurrent vowels[Assmann 89, Assmann 90], and this is similar to the steady-state music decomposition problem. Alain de Cheveigné at Université Paris 7 examined this problem by minimising the output power of two comb filters.[de Cheveigné 93, de Cheveigné 95, de Cheveigné 96] Tomohiro Nakatani at NTT developed a technique named HBSS, or Harmonic-Based Stream Segregation[Nakatani], and applied it to separating male and female speech.

5.3.23 Masataka Goto, Waseda, 1994

Goto at Waseda developed a system for separation of percussion instruments.[Goto 94] This uses an improved form of template matching. It separates a polyphonic mixture containing bass drum, snare drum, low/middle/high toms, open/closed hi-hat, ride cymbal, and crash cymbal, in the context of the near-simultaneous onsets that are common for percussion. There is a clear potential link with later MIDI-based work[Hidaka, Goto 96] on artificial jazz performance.

5.3.24 Mamoru Ueda, Waseda, 1994

Ueda examines decomposition of two concurrent sounds.[Ueda] If all the partials of each note have a scaled version of the same amplitude envelope, then his system can separate them. He shows good results from test data that met these requirements, but poorer results from actual instruments, and correctly deduces that the assumption may not be valid.

5.3.25 Jonathan Berger, Yale, 1994-95

At Yale University, Jonathan Berger and colleagues developed a polyphonic analysis system[BergerJ 94a, BergerJ 94c, Berger 95] using wavelets.[Coifman 90, Coifman 92] Their system has been applied to denoising old analogue recordings, including an 1889 recording of Johannes Brahms playing his Hungarian Dance no. 1 in G minor and Enrico Caruso singing an aria from Puccini's Tosca in 1903.[BergerJ 94b] Their wavelet-based system seems to work well at removing noise, although they report that some 'clicks and whistles' remain. Later work by Popovic extends this system to source separation and polyphonic pitch tracking of a vocal with piano accompaniment[Popovic 95a] and a solo piano excerpt with 5-note polyphony.[Popovic 95b]

5.3.26 Avery Wang, Stanford, 1994

Avery Wang designed a polyphonic system on the NeXT for source separation and related applications.[Wang] Wang's technique uses a Frequency-Locked Loop and frequency warping to implement tracking on the analytic signal. Four pieces are illustrated:- orchestra/*baritone*, orchestra/baritone/*soprano*, rock band/*vocal*, and harpsichord/trumpet/*vocal*, with the italicised voice being removed. His system implements resynthesis of the removed voice and of the accompaniment.

5.3.27 Douglas Nunn, Durham, 1994-96

In 1994 this author published a paper describing early results of the polyphonic transcription system described in the next chapter.[Nunn 94] Graphical applications are described separately.[Nunn 95] This system was applied to polyphonic piano and organ, didgeridoo, brass trio, and string orchestra. A multirate approach called Octave Spectral Analysis is used – this is virtually identical to Mont-Reynaud's BQFT (Bounded-Q Frequency Transform).[Mont-Reynaud 90]

5.3.28 Robert Höldrich, Graz, 1994-95

Robert Höldrich[Höldrich 94a, Höldrich 94b, Höldrich 94c, Höldrich 95] uses the Modified Moving Window Method[Kodera 78], based on the STFT, to analyse and resynthesise a multi-component signal.

5.3.29 Eric Scheirer, MIT, 1995-96

The system developed by Eric Scheirer[Scheirer 95a, Scheirer 95b] was designed for polyphonic transcription of piano music. In contrast to many of the transcription systems we have seen, but like Takami, this one uses the score of the music, and transcription is mainly directed at *expressive performance* extraction, determining the note onset times more accurately. It also estimates the offset times and the amplitude of each note. The onset times are extracted to an accuracy of better than 20 ms. He also extracts the offset and amplitude, although less accurately.

Scheirer raises this criticism of his own work: "It seems on the surface that using the score to aid transcription is 'cheating', or worse, useless – what good is it to build a system which extracts information you already know?". He contends that score-based transcription is a useful restriction of the general transcription problem. It can certainly be a useful tool for modelling cases where we *already* know the score, such as in listening to a familiar piece, where our musical cognition is directing us to *expect* notes. This system can also make use of timbral knowledge of the piano – as the piano is a non-driven instrument, the only feature being 'controlled' by the player after the onset of a note is the time of its offset. It looks for single notes by looking for increases in high-frequency (above 4 kHz) energy or overall energy, or by using a comb filter based on the target pitch. It looks for multiple notes by using multiple filters at frequencies selected so as to be (hopefully) unique to one note of the cluster.

Such a system is, one imagines, ideal when the score is known *and the player plays it accurately*. However, Scheirer notes that one potential example had wrong notes and unmusical phrasing and could not be considered.

His comments on polyphonic transcription warrant quoting:-

> A larger issue regarding the problem of general polyphonic transcription is the goal and motivations underlying them. Why is there so much interest in building transcription systems?
>
> We submit that it is for several reasons. Obviously, having a working transcription system would be a valuable tool to musicians of all sorts – from music psychologists to composers (who could use such a tool to produce "scores" for analysis of works of which they

only had recordings) to architects of computer music systems (who could use it as the front-end to a more extensive music-intelligence or interactive music system).

Another reason that so much effort has been invested in the construction of transcription systems is that on the surface, it seems as though it "should be" possible to build them, because the necessary information "must be" present in the acoustic signal. While this feeling seems to underlie much of the work in the area, it is so far drastically unjustified.

This point relates to a final reason, which is based on a hypothesis of the human music cognition system – that human listeners are doing something like transcription internally as part of the listening process. Stated another way, there is an implicit assumption that the musical score is a good approximation to the mid-level representation for cognitive processing of music in the brain.

It is not at all clear at this point that this hypothesis is, in fact, correct. It may well be the case that in certain contexts (for example, densely orchestrated harmonic structures), only a schematic representation is maintained by the listener, and the individual notes are not perceived at all. Since it is exactly this case that existing transcription systems have the most difficulty with, perhaps we should consider building transcription systems with other goals in mind than recreating the original score of the music.

Scheirer concludes, rather modestly, that "certain limited aspects of polyphonic transcription can be accomplished through the method of 'guess and confirm' given enough a priori knowledge about the contents of a musical signal". Later work[Scheirer 96] turns to replacing the score with 'high-level musical inference', such as musical rules describing four-part chorales.

5.3.30 Rolf Wöhrmann, Hamburg-Harburg, 1995-96

Wöhrmann and Solbach present a system[Wöhrmann, Solbach 96a, Solbach 96b] for tracking partials in white noise. They use a constant-Q wavelet analysis based on twelve gammatone filters per octave, and apply it to piano and mbira[55] music. Harmonics are suppressed by a phase locking algorithm, and notes are detected by thresholding. Their examples seem to show potential for ASA.

5.3.31 Thomas Stainsby, La Trobe, 1996

Stainsby concentrates specifically on polyphony, and works in non-real-time on a NeXT machine.[Stainsby] The aim is specifically source separation rather than transcription; no attempt to characterise the sounds are made. As with many other schemes, he uses a quasi-logarithmic frequency distribution, in this case from bounded-Q MQ (McAulay-Quatieri) analysis. As is also common, he then lossily compresses the data by fitting it to linear breakpoints. He then applies Bregman's grouping principles to form notes. His test piece has a polyphony of two, and appears to work effectively, but he notes that it is cannot yet assign all notes from one instrument into a single track. His system is monaural but may be expanded to stereo. The system is partly *interactive*, in that the user chooses many parameters before or during analysis. This is connected to earlier work by Scallan using MQ techniques in an analysis/synthesis package for the Macintosh.[Scallan]

---

[55] The mbira is an African thumb piano similar to a kalimba, with plucked metal tines attached to a resonator.

5.3.32 Keith Martin, MIT, 1996

Martin outlines the preliminary stages of a blackboard-based system designed for transcription of four-voice counterpoint played on a piano.[MartinK 96] This uses a shared workspace with several agents (or knowledge sources) operating quasi-independently.

5.3.33 Shawn Menninga, Calvin College, 1996

Shawn Menninga, at Calvin College, Michigan, developed a transcription system using the GFT (Generalised Fourier Transform).[Menninga] It has been tested for transcribing a mixture of oboe, flue, trombone, and double bass. In this system the instrument spectra are known in advance.

5.3.34 Other research

Other research not reviewed here includes work by Watson at Sydney[Watson], Heinbach at Munich[Heinbach 87, Heinbach 88], and Michael Hawley at MIT.[Hawley]

5.3.35 Comparison

The table below summarises most of the polyphonic systems above. EPA stands for expressive performance analysis. In many cases the aims fall into several categories.

Main author	Input	Poly.	Known score	Known orch.	Aim
Stockham	voice + orchestra + noise	n	?	?	separation
Parsons	2 × voice	2	n	Y	separation
Moorer	2 × guitar	2	n	Y	transcription
Chafe	piano, piano + harpsichord	n	n	Y	separation
Schloss	2 × conga	2	n	Y	transcription
Weintraub	2 × speech	2	n	n	echo cancellation
Mont-Reynaud	piano	2	n	n	transcription
Tsujimoto	ethnic music	n	n	?	transcription
Katayose	string, 2 of pno./gtr./clar./vln.	2	n	Y	transcription
Takami	piano	n	Y	Y	EPA
Kronland-Martinet	clarinet, trumpet, ?	2	n	n	resynthesis
Vercoe	?	n	n	n	transcription
Maher	2 × synth, clar.+bsn., tpt.+tuba	2	n	n	transcription
Pearson	piano, woodwind trio	2-3	n	n	transcription
Doval	2 × clarinet	2	n	n	transcription
Cooke	speech + various	1 + n	n	y?	separation
Brown	speech + noise	1 + n	n	y?	separation
Ellis	environmental sounds	n	n	n	transcription
Kashino	flute/clar./tpt./vln./pno.	2-3	n	n	separation
Pressing	?	?	n	n	transcription
Nakatani	male speech + female speech	2	n	n	separation
Goto	percussion	n	n	y	transcription
Berger	piano + noise	n	n?	y	denoising
Wang	voice + various	1 + n	n	?	separation
Nunn	various	n	n	n	transcription
Höldrich	synthetic	2	n	n	separation
Scheirer	piano	n	Y	Y	EPA
Wöhrmann	piano/mbira + noise	n	n	n	separation
Stainsby	?	2	n	?	separation
Martin	piano	4	n	Y	transcription

Table 16 - Summary of transcription systems.

The design of a system depends largely on the intended application. A note transcription system can be designed for a particular instrument to achieve robust time performance, and for auto-accompaniment and expressive performance analysis, the score may also be known. In contrast, applications to documentation of electroacoustic music can assume nothing about the input. For auditory modelling and beat tracking, analysis is the primary concern, but for compositional use, transformation and synthesis are vitally important. For compression, source separation is unnecessary, but in psychoacoustics it is the central concern. Live performance systems require real-time performance, but complete analysis/resynthesis systems may take 36000 times longer.

5.4 Other systems

Several other systems described below do not quite fall under the category of 'automatic music transcription machines'.

5.4.1 Manual/interactive systems

Several of the methods described above are partly interactive.[Mellinger 91a, Mellinger 91b, Tsujimoto, Crawford, Pressing 93a, Stainsby] There are more interactive methods that rely as much on computer tools than on computer analysis. At the extreme end, it is possible simply to examine each onset separately with a graphical waveform editor and/or a spectrogram. By zooming in on each note, and auditioning short segments, it is possible to determine the onset time to a reasonable degree of accuracy (although detecting near-simultaneous onsets is still problematic). This technique was used by Bruno Repp in his analysis of 28 performances of Schumann's "Träumerei".[Repp] His analysis and my attempts to repeat it are described in chapter 7 and Appendix K.

5.4.2 Multiple-channel systems

The difficulty of transcription might prompt the suggestion that if one wanted the sound of each instrument in a brass quintet, one should have recorded them using five microphones. However, unless each instrument has its own acoustically isolated room, each microphone will pick up sound from other instruments. Many similar situations arise in speech applications, and source separation is often referred to as *blind separation* or *blind deconvolution*. Mitchell discusses what is often termed the 'cocktail party effect', and presents a system for separating two voices using four microphones[Mitchell], and Chan examines the process of separating N instruments recorded by N microphones.[Chan] Bell uses a neural-network approach to separate up to ten voices.[Bell] Mansour shows that it is also possible to separate N sources *convolved* with each other.[Mansour]

These methods of source separation make use of sound localisation. If several sources are fixed in position with respect to more than one channel, it is possible to separate the sources. A simple example of this type of source separation is voice removal for karaoke features on some home stereo systems.[Technics] This assumes that the vocals are central in the sound stage, which is often true, and removes the component common to both channels. Filtering before and after ensures that this does not remove central low frequencies such as the bass drum and bass guitar. This process is crude but can be effective. A refinement of this process is found in the Thompson Vocal Eliminator™, a commercially available device to remove vocals from music.[LTSound] The manufacturers report that vocals are virtually inaudible for 25% of examples, and barely audible for another 25%. However, if the voice moves in spatial position or is heavily processed then this method cannot be used.

5.4.3 Beat induction systems

Beat induction, or beat tracking, or foot-tapping, systems are a special case of transcription where the only desired output is the temporal structure. The 1994 ICMC held a special session on such systems. An overview and an extensive bibliography is given by Desain and Honing.[Desain 94] Tait also argues for concentration on the large-scale time domain, and it seems reasonable to suggest that understanding the rhythmic structure will be of help to a transcription system, as it will allow more intelligent guesses about where notes might or should be.[Tait] Analysis of rhythm is also examined by Todd, who uses a wavelet-based approach to decompose the sound energy flux of polyphonic piano music into a grouping structure and a metrical structure.[Todd]

5.4.4 Chord induction systems

Bernice Laden describes a neural-network approach to classifying chords as major, minor, or diminished.[Laden 91] Neural networks entail the force-feeding of the net with lots of questions and answers, then testing how the same or similar questions can be answered by increasingly smaller nets. Her later work turns to neural network recognition of pitch, and her system is able to recognise polyphony of 2 and 3.[Laden 94] Such systems carry out a very lossy analysis as they do not represent timbre, and the approach could probably not be extended to polytimbral note transcription, as there would have to be output units for *each* timbre – for a hypothetical wave-to-General-MIDI system this means 128×128=16384 output units. This would be unwieldy, and would still omit amplitude and other timbral information. Moreover, the training data is not available for new sounds.

5.4.5 Speech-based systems

Work by Parsons, Weintraub, Cooke, Brown, de Cheveigné, and Nakatani has been described above. Several other systems[Hanson, Naylor, Childers, Amuedo, McAdams] designed purely for speech signals are described by Barry Vercoe.[Vercoe 88]

As with music, analysis of a single source, while not trivial, can be accomplished satisfactorily, but separation and analysis of several sources is much more problematic. A common application is speech enhancement, where one source is to be made more intelligible and other sources are to be removed or made less intelligible.

Systems designed *exclusively* for speech may make so many assumptions about the input so as to be ineffective for music. Moreover, many speech applications only require intelligibility, and accept a lower fidelity than musical applications. Yet we cannot lay a boundary between speech and music; a system sufficiently general as to handle vocal music should also be able to apply the same principles to speech.

5.4.6 MIDI-based systems

If we are concerned with studying a keyboard performance, one obvious possibility is to record the performance using MIDI. Also, some auto-accompaniment and artificial performer systems assume that

the information is already in MIDI format.[Pennycook 93, Hidaka, Goto 96, Baird] Obviously this is only possible if the performer is still living and is willing to specially record a performance. It also requires a high-quality MIDI keyboard with a sufficiently similar mechanism to a concert grand piano (and an acceptable synthesis method), or a piano with MIDI output such as a Yamaha Disclavier[Scheirer 95b], or a Bösendorfer optical recording piano.[Palmer]

5.5 Summary

Many have attempted polyphonic transcription, and while each approach has demonstrated some success, none appears to be sufficiently general. A common feature of most systems is a hefty set of restrictions placed upon the input – few are at the stage where they can be plugged in between a radio and a printer. The issue of robustness is discussed by Dixon and Sterian.[Dixon, Sterian] As Dixon points out, "most approaches suffer from *brittleness* – a steep degradation in performance under non-ideal conditions". There is no standard corpus of test pieces for transcription, and there are difficulties in quantifying the accuracy, so it is in general not possible to directly compare performance. Also, the large amount of data in the original audio and in anything derived from it limits many examples to a few seconds.

The next chapters present my attempt at polyphonic transcription. It is designed with the intention of being applicable to a variety of music.

6. Design of the transcription system

In this chapter I introduce the design of my analysis and transcription system, and describe each of its components in detail.[Nunn 94]

6.1 Specification

As discussed in previous chapters, the term 'transcription' has two meanings. It can refer to the process of analysis and classification that would be required to determine the (MIDI) score data. This is a worthwhile aim in itself. However, it is a lossy process and does not permit resynthesis of timbre. When viewed in the context of the overall analysis-transformation-resynthesis paradigm, the output we desire is the complete set of control information needed for resynthesis by the complementary method. The system I will describe is intended primarily for this latter purpose. MIDI transcription is a useful by-product that also allows a method of evaluating the system performance.

While some of the systems reviewed before use a physiologically accurate model of the human auditory system, the model here will not make an attempt to do so. As Wang points out, our physiology gives us a blind spot in our vision but we are normally unaware of it.[Wang] It would surely be counterproductive to implement this in a computer vision system.

In the model I describe in this chapter, the basic sonic entities are sinusoids multiplied by a rectangular or other envelope. These sinusoids are to be grouped according to which partial of which note they belong to, although this is not strictly necessary for resynthesis.

The system is designed for monaural audio, a reflection of the fact that separation can be carried out on a monaural signal.

6.2 Overall design

The transcription system[Nunn 94] has several components, as shown in Figure 39. Neither the PC nor the C40 platform has sufficient resources to allow all of these stages to run concurrently. Instead these components are separate programs. The initial stage of analysis is carried out on the C40 and produces a large output file. The later stages are carried out on the PC alone, and in a similar way 'communicate' by passing large files to each other.

The original input files are either created digitally or recorded using the GoldWave[Craig] or Cool[Syntrillium] Windows sound editors.

6.3 The MEX script language

There are many stages running as separate programs, as mentioned above. It is possible to partly automate the process using a DOS batch file.[56] However, a change in one of the programs would then mean that all such files had to updated. Thus, a script language called MEX (Music EXecutor) was developed to integrate the various components of the process. This also allowed experiments to be scheduled to run unattended, often overnight.

The script language implemented by the C program MEXEC is similar to a batch file but also allows variable assignment. Variables, which are all text strings, are assigned using `Variable=Value` and referenced by `!Variable`. Variables are generally concatenated with whitespace between, and can also be concatenated without whitespace using the operator `!&`. Numeric variables are not implemented.

Lines can be numbered, as in BASIC. There are currently only two program flow commands – `GOTO` and `IF...GOTO`.

```
IF !Machine=Dan GOTO 20
...
20 CWDir=D:\mex\
GOTO 29
```

Other commands include:-

```
; semicolons start comments
```

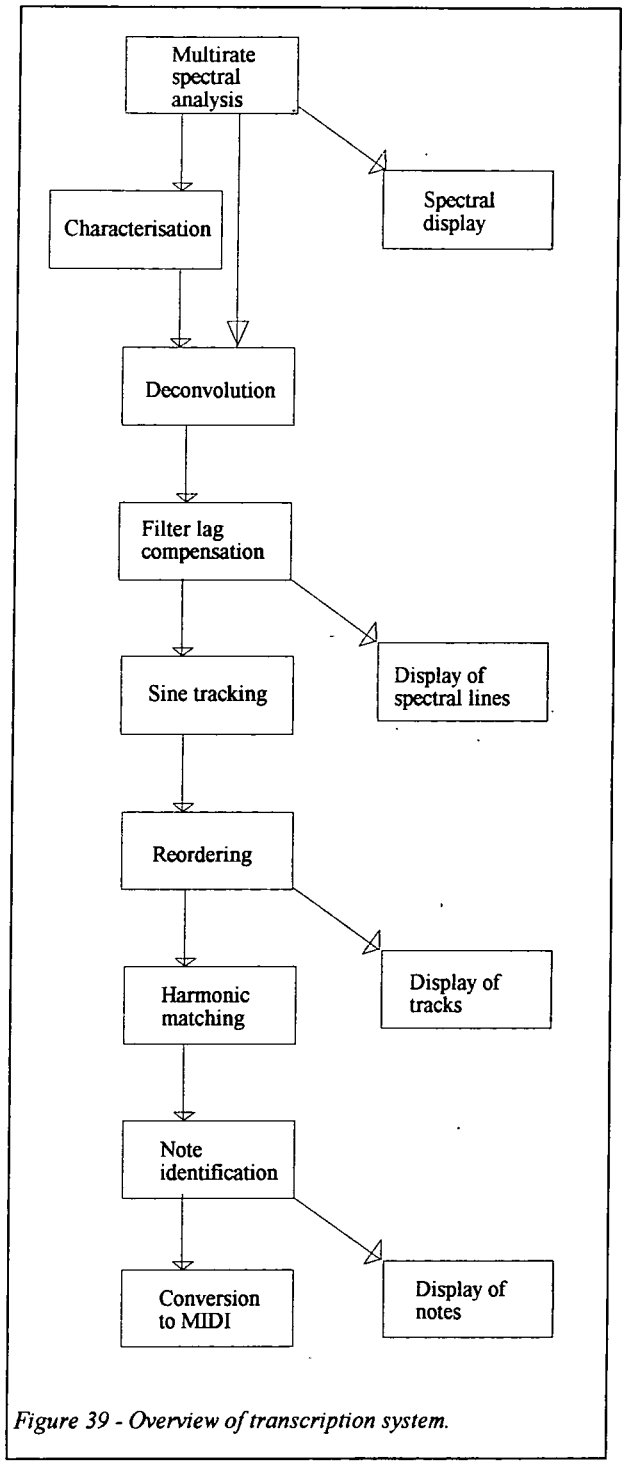


Figure 39 - Overview of transcription system.

[56] Batch files were still used on some occasions, such as when running one of the programs many times with different parameters, rather than each of them once.

```
ECHO Hi there, user.                        ; Print a message to the user
EXECUTE !DisplayProg !MonoFile !Flags       ; Run an executable file
DOS mode con lines=50                       ; Do a DOS command
PAUSE                                       ; Pause
END
```

The complete syntax is formally presented in Appendix E and a listing of the script file used in the analyses is given in Appendix F.

6.4 Multirate spectral analysis

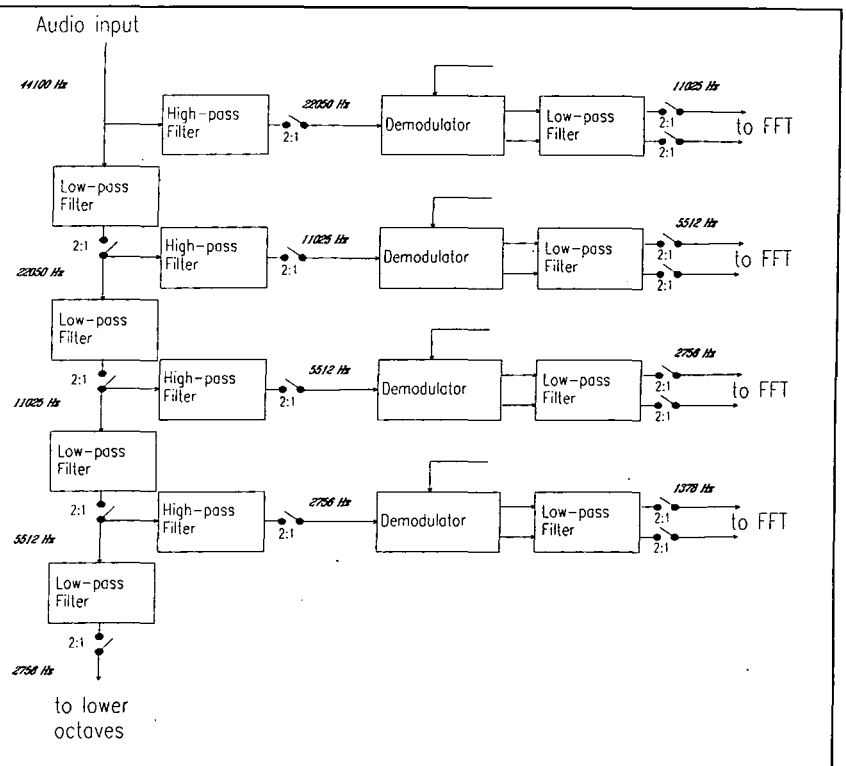


Figure 40 - Octave Spectral Analysis.

This process takes place on the TMS320C40, described in an earlier chapter.

6.4.1 Octave Spectral Analysis

The principle of Octave Spectral Analysis (OSA), shown in Figure 40, is to treat each octave[57] separately. There are several methods of doing this; the method described here is based on the method described by Elliott and Rao.[Elliott] If the sample rate is taken to be 44 kHz for simplicity, then we first use a high-pass[58] filter to attenuate all but the highest octave (11-22 kHz). We then decimate by a ratio of 2:1 – i.e. every second output of the filter is discarded. This aliases the signal down to the range 0-11 kHz. This is then demodulated by multiplication with real and imaginary sinusoids at 16.5 kHz, which brings the region of interest down to ±5.5 kHz. A low-pass filter then further attenuates the frequencies outside this range, which only represent the complex conjugate of the desired band, allowing a further 2:1 decimation, and then the FFT is carried out. The

[57] This technique can also be used to split the range into divisions other than octaves. However, the octave is by far the most convenient for musical and other applications.

[58] Elliott and Rao used a band-pass filter, as their application was not concerned with the entire range of input frequencies.

octave below this is treated in the same way, but at half the sample rate. There is little point in staying at the 44 kHz sample rate in order to analyse lower frequencies. Thus, each octave requires only half the computation of the one above, leading to a total computational load equal to 1+½+¼+... = twice the computation of the highest octave.

This gets us closer to the goal, but the frequency bins in each octave are still related by absolute frequency, and not truly logarithmic. A possible way to turn these into bins based at equal pitch intervals would be by interpolation. However, it is unlikely that the computation involved can be justified. Moreover, we would need to assume equal temperament and middle A = 440 Hz, neither of which is guaranteed. Furthermore, the harmonics (notably the seventh, which is around a third of a semitone flat) would still not fall precisely into keyboard-based bins.

The Q is narrower at the low end of the octave, and varies within a factor of 2 – but the values are the same across octaves. The improved pitch resolution at the bass end is at the expense of poorer time resolution, and the FFT size and choice of window determine where this trade-off is made. Initially, the analysis used an FFT length of 64, but later other sizes were implemented and compared. This is described in a later section.

A fundamental decision must be taken at the outset – choosing a fixed sample rate. A high sample rate means more memory is required, and does not necessarily mean better performance. A 440-Hz sine will end up in the 250-500 Hz octave, whether it was originally sampled at 16 kHz or 32 kHz. The sample rate must, as always, be higher than double the highest frequency in the signal. Inevitably, considerations of memory, speed, and mass storage space for both the input data and the various output files impose limits on the sampling rate and/or the length of the piece.

The FFT generally uses a Hamming window. This gives cleaner spectra, but broadens the peaks. Other choices of window are possible, and are discussed later.

The FIR filters are half-band filters, and the cutoff is at quarter of the sample rate. The high-pass output represents the top octave, 11-22 kHz. We can now discard every second sample, which aliases the range down to 11-0 kHz. The FFT is carried out on this (reversed) octave. The low-pass output, 0-11 kHz, is also calculated, and we again discard every second sample and use this as the input to the next branch. Each branch carries out the same procedure at half the sample rate of the one above. There are eleven octaves in all, which allows us to recognise frequencies down to 10.8 Hz at a sampling rate of 44100 Hz (or 11.7 Hz at 48 kHz). We note here that frequencies near the boundary between octaves could be represented in both octaves.

Since the filters are non-ideal, there will be aliasing between octaves. Energy still remaining in the stop band at 10 kHz will be combined with the 12 kHz component. However, this is also true at the resynthesis stage. When certain conditions on the two sets of filters are met, these two errors will cancel out exactly. This is known as the *perfect reconstruction* property. Quadrature Mirror Filters (QMFs)

and related filters can provide this. It was first proved for our case of two banks[SmithM 84, SmithM 86, Vetterli, Mintzer,] and has been generalised to N banks.[Chu, Vaidyanathan 87, Vaidyanathan 90] It also generalises to multiple dimensions.[Shah]

6.4.2 C40 task arrangement

Parallel C allows separate tasks, and the most effective way to design a large and relatively complex program is a modular approach. In addition, a long-term aim has been to look at the feasibility of a larger network of C40s. Thus, the multirate analysis was split into two separate tasks connected by a channel. The *driver* task handles the connections to the host PC, such as file and console I/O. The *analyser* task handles the computation – in this case the Octave Spectral Analysis which includes filtering and FFTs. Eventually, these may be on separate processors.

This raises a practical problem – if the second processor is not directly connected to the PC, how will it report debugging and other messages? The version of Parallel C in use at the start (1.0.0) did not permit I/O from remote processors. The simplest solution was to add a third task, `newsdesk()`, which is on the first processor, and is connected to all other tasks by channels. This task simply prints out everything it receives. Each task uses a standard protocol to send a string to the newsdesk. The command `say("hello")` will send "hello" on the specified channel to *newsdesk*. To show which task sent the message, it is more useful to use the command `isay("hello")`, which will output "Analyser: hello" to the screen.[59] Figure 41 shows the arrangement of the three tasks.

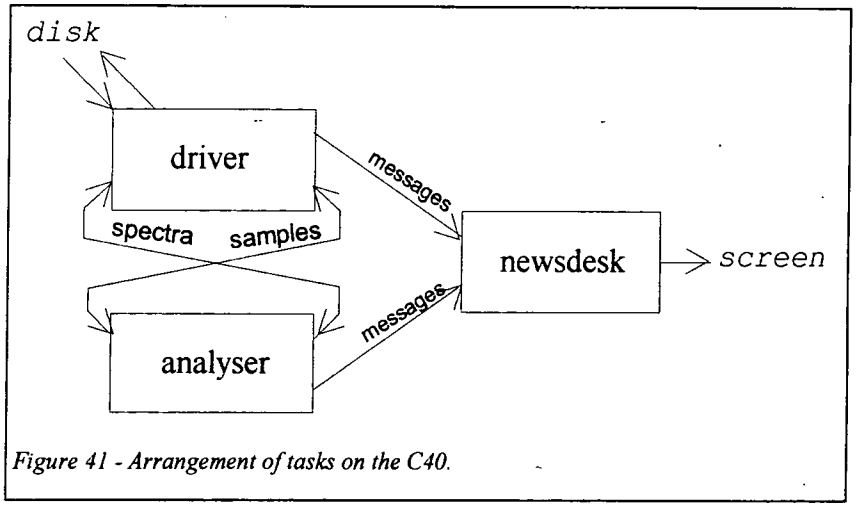


Figure 41 - Arrangement of tasks on the C40.

Since there may be more tasks than processors, there is an overhead in switching between tasks, due to saving and loading registers. At one extreme, we could read bytes individually from disk and send them, and the analyser could read them individually. This actually means that the processor must switch between tasks every byte. It makes more sense to read from disk and to transfer data in larger packets. The other extreme also has problems, in that if the intermediate buffers are large, we will not be able to derive any analysis results until long after the input has been presented. To evaluate where the

[59] Later versions of Parallel C addressed this shortcoming, and allowed any task on any processor to print directly to the screen. The kernel automatically routes messages via the root.

compromise should be made, the performance was measured for varying sizes of disk buffer and channel buffer. These results are shown in the next chapter.

6.4.3 OSA thread arrangement

The analysis of each octave is handled by the same routine running eleven times, and the scheduling of these processes is very straightforward. The process is exactly the same in each octave; although the filters are at different frequencies, they are the same with respect to the frequency of the incoming data. Thus the same program can be used for all octaves. This is very efficient on a multi-threaded system; each of eleven threads runs the same program, so there is no duplication of program code.

Figure 42 shows the threads in the task OSA.C. There are thirteen threads running concurrently; eleven for the eleven octaves, the `main()` task which reads the data from the driver task, and `womble()`, which "picks up the pieces and makes them into something new" by collecting output buffers from all threads, carrying out the FFTs, and sending the results back to the driver. The driver and the newsdesk tasks, in the initial implementation, only have one thread each.

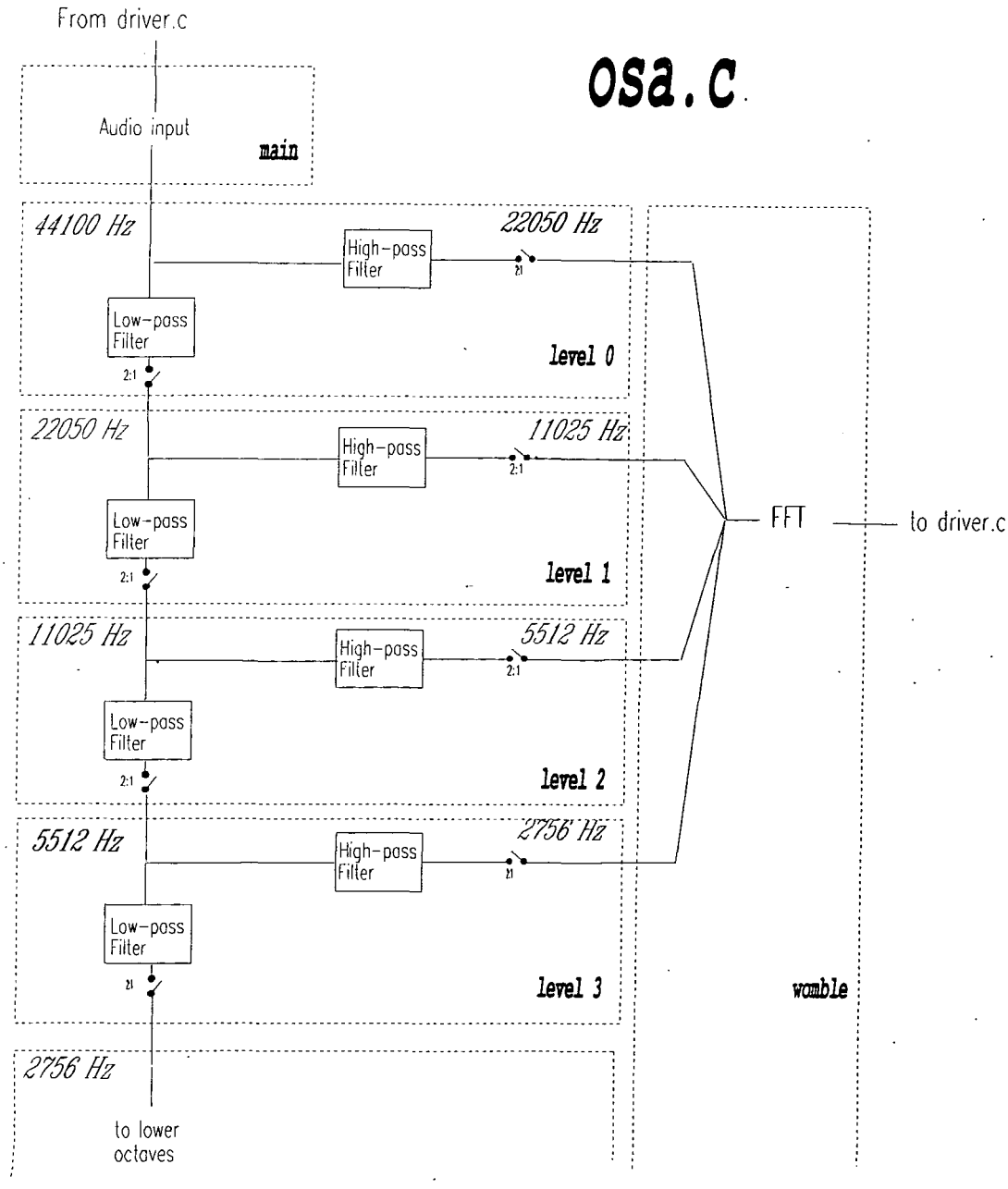


Figure 42 - Threads in OSA.C.

6.4.4 Filter specification

The perfect reconstruction property implies that sharp filters are not an absolute necessity. However, it is still better to use very sharp filters, as we want to recognise the frequencies without undue distortion or aliasing. Also, the errors can build up across octaves. The low-pass filter output is calculated by subtracting the high-pass output from the original. Thus, the passband ripple in one is the same as the stopband ripple in the other. For optimum use of filter taps, the ripple in the passband should equal the attenuation in the stopband. This is satisfied[SmithM 86] with a Kaiser window.[Kaiser 74]

For applications not requiring resynthesis, IIR filters can be used. These give a shorter filter for equivalent performance, but distort the phase of the signal. For this reason it is not possible to achieve perfect reconstruction with an IIR filter unless costly all-pass phase-equalising filters are used, and filter stability is said to be problematic.[SmithM 85, Swaminathan, Jaw]

The FIR filter (FIRK6.FLT) used has a length of 255. It was designed using DFDP, the Digital Filter Design Package.[Atlanta] Figure 43 shows the frequency response, and Figure 44 shows detail of the band from 10 to 20 kHz. The filter impulse response is shown in Figure 45.

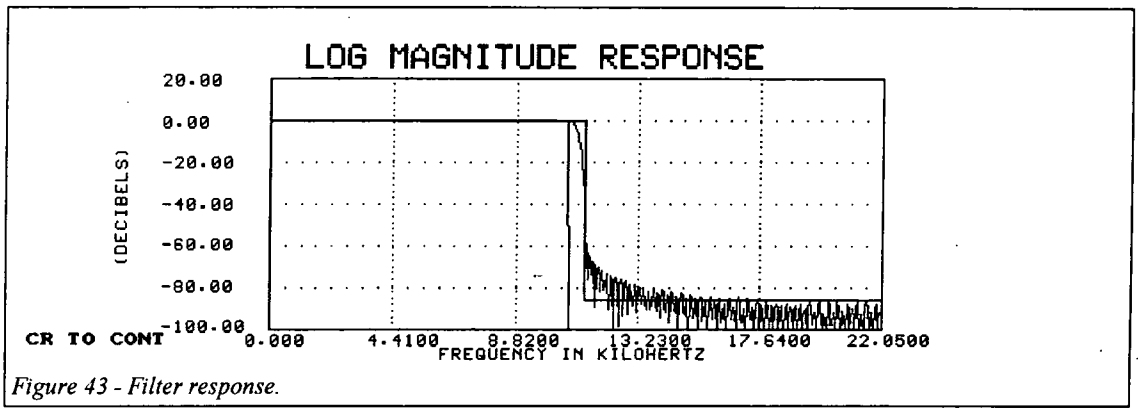


Figure 43 - Filter response.

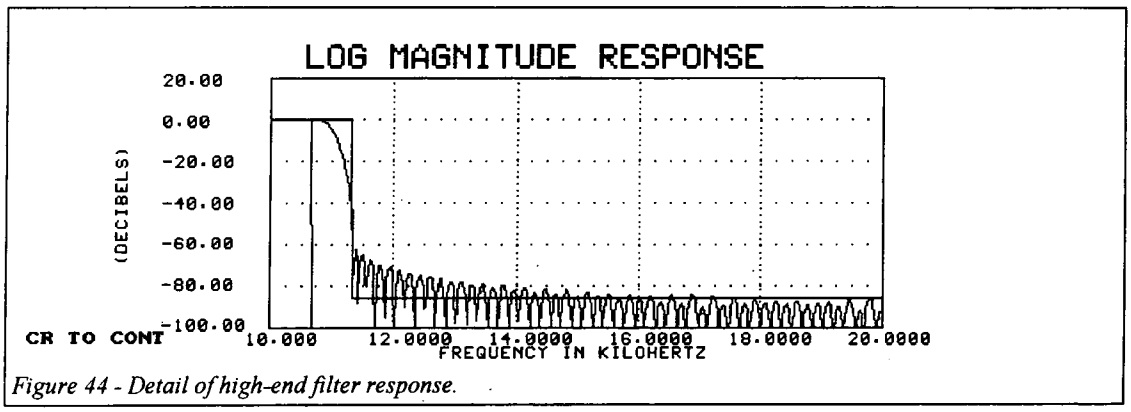


Figure 44 - Detail of high-end filter response.

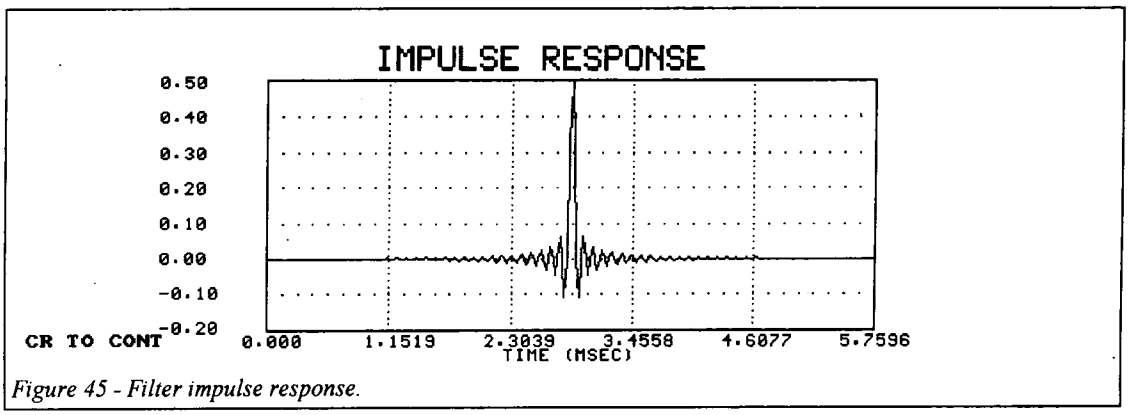


Figure 45 - Filter impulse response.

Another filter was designed (FIRK5.FLT), with a length of 511, and could be selected through the setup menu. However, this is probably an unnecessarily high computational cost for the sake of slightly more accuracy.

6.4.5 Filter algorithm

There are several simplifications that we can take advantage of. The HPF and LPF need not both be computed:- we get the LPF by subtracting the HPF output from the delayed original. Also, every second filter coefficient, expect the middle one, is zero, which halves the computation. Third, the filters are symmetrical. Fourth, there is no point in calculating the samples that are to be discarded. All of these allow a factor-of-2 reduction in the computational requirements. The filters are implemented as polyphase filters[Bellanger], as shown in Figure 46.

$$y7=0.5*x7+a0(x8+x6)+a1(x10+x4)+a2(x12+x2)+a0(x14+x0)$$

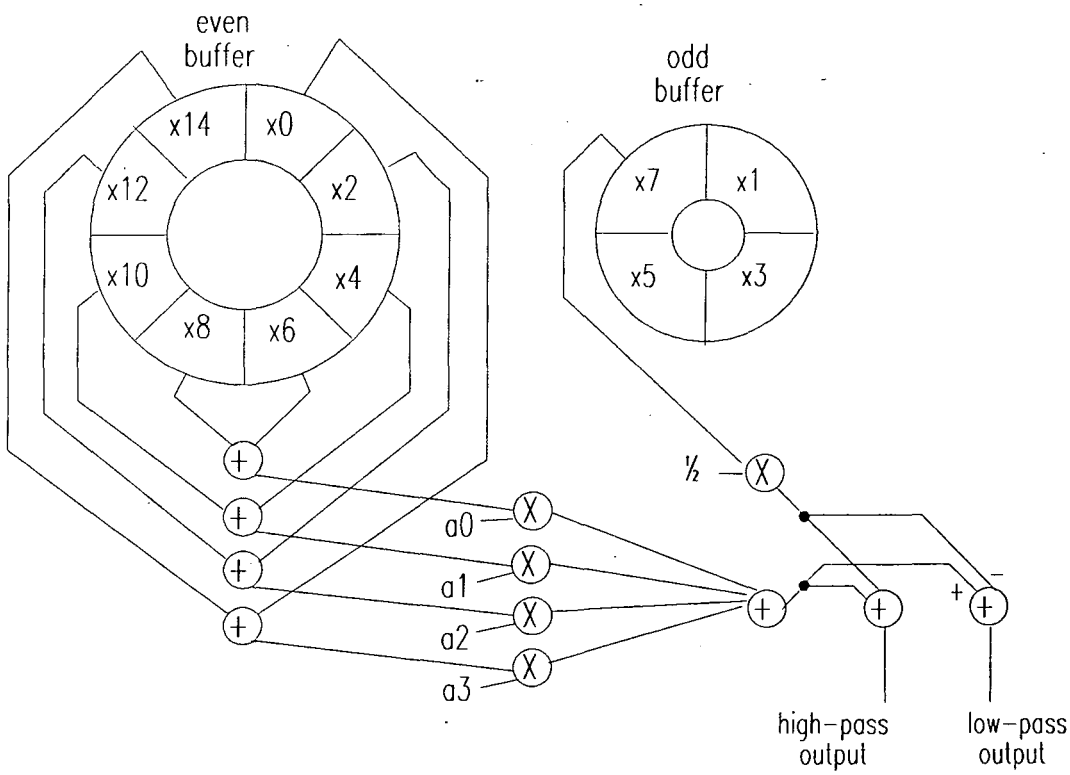


Figure 46 - Schematic of filter implementation.

6.4.6 Filter timing

The filter length of 255 means that there is a group delay of 128 samples. However, since the sample rate decreases for lower frequencies, the group delay gets progressively longer. One alternative is to stop decimation and leave a 'last' band from 0 to, say, 100 Hz. This approach is advocated by Phillips, whose additive synthesis engine is designed for real-time use.[Phillips 94, Phillips 96] However, the current system does not adopt such a scheme. The eleventh octave holds the FFTs for 10.77 to 21.53 Hz

(assuming a sample rate of 44100 Hz), and the subsonic range from 0 to 10.77 Hz is written as a 'residual block' into the file. This ensures that reconstruction would be possible, although in practice these blocks are ignored by later programs.

6.4.7 FFT size

In the original implementation, the FFT buffers are of size 64. A length of 2^N allows use of the FFT rather than the DFT. However, there is no reason to suggest that the sample will have any correlation with this size, or with any integral periodicity. The choice of FFT length determines the trade-off between time and frequency resolution, and is discussed in more detail below.

The FFTs with lengths of 64 or more were implemented using highly optimised assembler source, supplied by Texas Instruments. This routine makes use of the C40's ability to address a memory block in *bit-reversed* order. However, it was unable to handle FFTs shorter than 64, so for this, public-domain C source code based on FOUR1 and REALFT in 'Numerical Recipes' was used.[Press]

A further extension could be to apply more intelligent segmentation to the stream of each octave, and to allow any integral block size. This would of course lose the ability to use the FFT – DFTs are less efficient.

6.4.8 FFT buffering

Within each octave, the processing load fluctuates with time. When the FFT buffer is full, the FFT is calculated and output. Thus there is a sudden demand for computational power every 64 samples (if 64 is the chosen FFT size), i.e. when the time is divisible by 64. If we consider all the octaves, we see that there is a similar demand added when the time is divisible by 128,256,512,..., and the total demand follows a profile analogous to the divisions on a ruler, as shown in Figure 47.

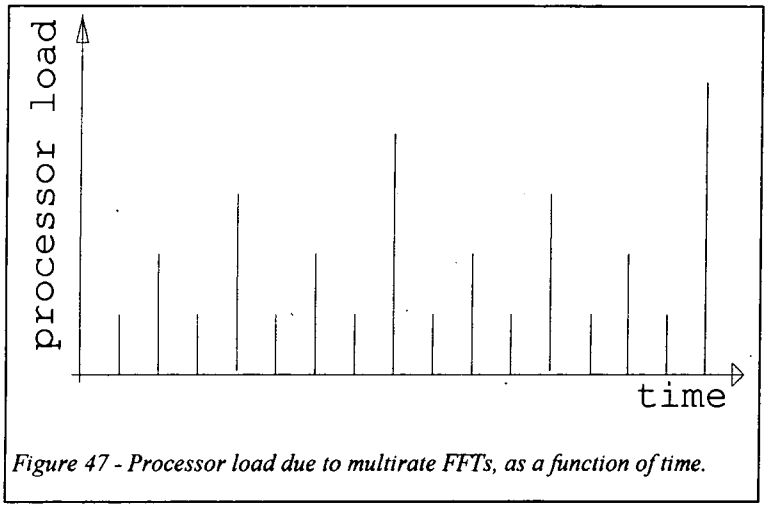


Figure 47 - Processor load due to multirate FFTs, as a function of time.

It is possible to spread the FFT computation evenly across the input block, so that instead of performing O(N×logN) calculations at the end of the block of N, we perform O(logN) calculations every sample. This is described in Appendix A. With this, the processor load is much smoother. The spectrum of one block is calculated and output, one bin at a time, *while* the next block is being input. The main disadvantages of this technique are that it returns the spectrum slowly, and that it requires temporary storage space for its partial results; an extra N-byte block of memory must be set aside for this. It

remains to be determined whether this technique solves more problems than it creates, so this has not been implemented.

6.4.9 Summary of OSA

Octave Spectral Analysis offers an approximately constant Q, and can be implemented efficiently on the C40. Despite the powerful environment, though, this analysis runs slower than real time. It could reach real-time performance with shorter filters or with more processors, but the overheads of reading and writing to a disk are ever-present. There is also a latency corresponding to the group delay of the filters, which is several seconds for the eleventh octave. (Most of the following processing is implemented on a stand-alone PC, and takes several hours, so this is not the most time-intensive stage of transcription.)

6.5 Spectral display

The multirate analysis produces a large output file (*.C40). It is twice the size of the original – there is the same number of data points, but whereas the original data was in 16-bit words, C40 words (whether `char`, `int`, `float`, or `double`) are 32 bits long. There are also a few bytes added at the start of each block, plus a global header, so the total size is around 2.08 times the original size.

The next stage is to examine the file to ensure that the analysis is correct. This is done using graphical display. The graphical output described below and at later stages proved to be an essential part of the system, both for presenting the results and for debugging the system itself.

6.5.1 Display techniques

As the C40 has no support for graphics, the graphical displays are done by a standalone PC program called READSP (READ SPectrum) which graphically plots the analysis file. The resolution was limited to the standard VGA resolution of 640×480, partly due to the complexities of supporting SVGA graphics, and partly due to the lack of printer support for higher resolutions. Comparison of the performance on several PCs confirmed that fast graphics hardware is advantageous.

The display is essentially a spectrogram, but for our multirate system we are plotting according to bins that are *roughly* equidistant in pitch. A typical spectrum is shown in Figure 48. Here, half a minute of sound is displayed on the screen, but a more typical use is to expand the time axis by scrolling the display. The horizontal axis is log-frequency; the vertical axis is time, from top to bottom. The eight boxes show the colour scheme, and the treble and bass staves are shown next to this. Throughout this chapter, I use the Mendelssohn to show examples. The analysis of this and all other pieces are described in more detail in the next chapter.

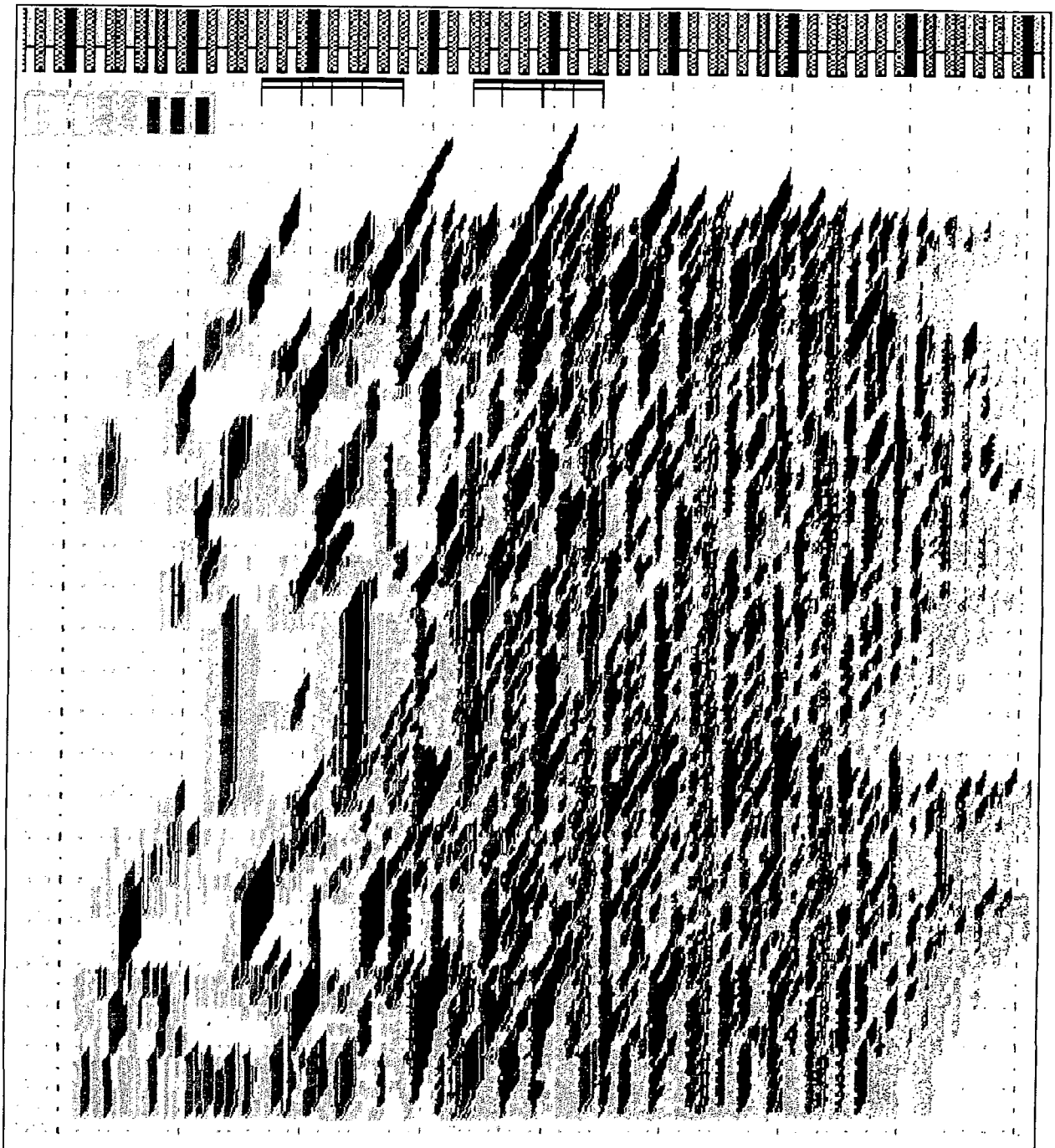


Figure 48 - Logarithmic spectrogram of 30 seconds of Mendelssohn's Sonata 3 for Organ.

Two axes are needed for time and log-frequency, so amplitude may be plotted using a third 'dimension'. This is roughly north-west, but has the appearance of pointing upwards. The height of each block is proportional to the amplitude of the transform bin. However, amplitude may vary over many orders of magnitude, so the colour of the block is also used to denote the logarithm of the amplitude.

The example above plotted time from top to bottom and frequency from left to right, but it is more conventional for time to go from left to right and frequency from bottom to top.

The displays can be produced in a range of styles. In the 'skyscraper' display shown above, the magnification and angle of the blocks can be varied. The display also allows seven colour schemes – mono, 8-colour, 16-colour, 8-grey, 16-grey, blue/yellow, and 'fire', and any of these can be reversed.[60] These displays are shown separately in Appendix N.

6.5.2 Animation techniques

Animations can be created as part of the compositional process.[Pringle] However, here there is an opportunity to derive them from the sound itself, based on the spectral displays described earlier. A screen-capture TSR is loaded first. Then, while the display program is running, screens are periodically captured, and afterwards these are compiled into a movie format.

Two such screen capture programs were used:- VGACAP[Gozum] and PCXDUMP.[Frandsen] VGACAP must be operated manually, by pressing its hotkey, so provision was made to pause the display whenever the screen was about to scroll (usually rightwards, occasionally downwards). VGACAP saves the screen as a raw file plus a palette file. Since the former is large, and the collection thus formed might otherwise fill the entire disk, the display program has the facility to call a shell file which uses PKZIP[Pkware] to add them to a single compressed file. VGAFIL is used to form a *.GIF file from them. One annoying bug prevents it from saving the palette colours correctly. PCXDUMP can be installed so as to be triggered by calling a user-specified interrupt, thus removing the need for the user to press a hotkey hundreds of times. It compresses files into the PCX format immediately. Due to these factors, PCXDUMP was found to be more useful.[61]

To create the film format, there are again two possibilities:- DTA[Mason] (Dave's Targa Animator) and VFD[Williamson] (Video For Dos). DTA accepts PCX/TGA/BMP pictures, but not GIF, and produces FLI/FLC movies. The FLI format is the 320*240 format used by Autodesk Animator; FLC is an extension to 640*480; FLH and FLT formats are similar but have 16-bit colours and 24-bit ('true') colours respectively. VFD accepts TGA pictures, but not GIF or PCX, and produces AVI format movies. DTA was found to be the more useful of the two.

The animations are displayed using either QV[Hesseler] (QuickView) or DFV[Mason] (Dave's Flic Viewer). As the screen is scrolling, there is quite a large difference between successive frames, and the FLC format is inefficient. Both viewers (on a 66-MHz 486DX) were capable of a frame rate of around 3-4 frames per second. While this is jerky, it is still effective.

One command-line flag of DTA holds interesting possibilities. The flag `/3D` causes DTA to read two sets of files, taking them to be the left and right pictures for making a 3-D film. The two sets of pictures

---

[60] The 'reverse' flag, in conjunction with a mono or grey-scale scheme, was used for many of the screen shots in this and the following chapter.

[61] Both programs were also used to capture many of the figures in this thesis.

are made in the same way, but with a slightly different slant angle (and a different magnification to compensate for this). The resulting animations, when viewed through a pair of 3-D glasses[62], convey the height as 'standing out' from the screen.

6.6 Characterisation

The next program, called DISTRIBx[63], also reads the files produced by the Octave Spectral Analysis and plots a histogram of the energy separately for each octave. It also prints out a characterisation file, which consists solely of a single figure giving the 'average' energy level. This is done to normalise the processing of pieces at different overall intensities.

The display produced is shown below in Figure 49. It shows the distributions for each of the eleven octaves. The actual output is in colour to distinguish the eleven plots.

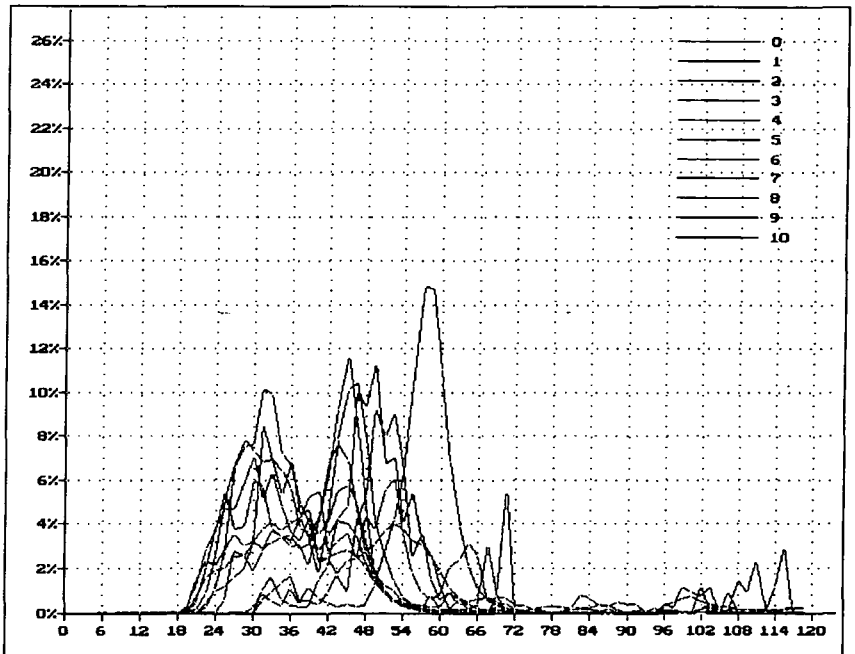


Figure 49 - Screen shot from DISTRIBx.

6.7 Deconvolution

Once we have the spectra from each octave, we then try to determine the sinusoids contained in it. This procedure is either imprecise or computationally expensive, or somewhere in-between, in that there is no quick way to calculate it accurately, and several ways of approximating the real spectrum, such as AR (AutoRegression)[Marple, Therrien, Foster], MA (Moving Average)[Marple], and the hybrid ARMA.[Marple]

---

[62] Several pairs of red/blue anaglyphic 3-D glasses were kindly supplied by American Paper Optics of Memphis.

[63] Here 'x' is the version number, so the name of version 2 is DISTRIB2.EXE.

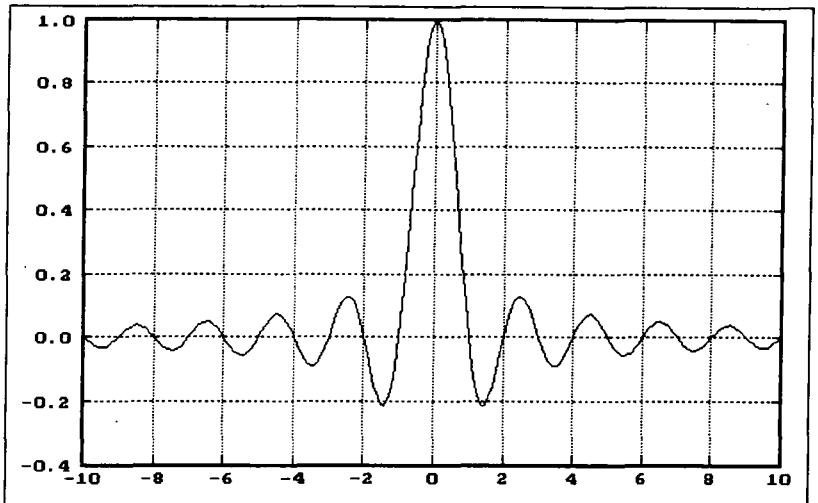


Figure 50 - sinc(x) versus x.

6.7.1 Theory of deconvolution

A major difficulty in applying spectral analysis is that we have only determined the spectrum at a small discrete set of frequencies. However, the spectrum actually contains frequencies between the bins, and it is necessary to try to determine these. To simplify the following discussion, all frequencies in the following paragraph are normalised with respect to the analysis frequency. If there is a frequency of 30.9 (times the analysis frequency), then most of its energy will go into frequency bin 31, but some will 'spill' into bins 30, 29, 28, ... and 32, 33, 34, ... The worst case is when the frequency is halfway between bins, in which case 30.5 would be represented by equally strong peaks at 30 and 31, with considerable energy in 29/32 and 28/33. The actual amplitude response of any bin to any frequency is given by a sinc-shaped curve, where sinc(x), illustrated in Figure 50, is defined as $\sin(\pi x)/(\pi x)$.[64] In other words, if there is energy m_a at f_a (in units of bins), then it will appear in bin f_b with an amplitude of $m_a \times \mathrm{sinc}(f_a - f_b)$. There is also a linear phase shift to take into account.

6.7.2 Implementation

The deconvolution is carried out by a program called PICKOUT. The approach used here is to assume that the signal is close to a sum of sines, and to try to deconvolve the spectrum directly. The method used assumes the spectrum to be fairly well-behaved – i.e., that there are only a few clearly separated peaks. This may not be true, of course, so the following procedures will have to be fairly robust.

We can define the 'fit' between the spectrum near f_o and the expected sinc response by calculating the correlation between them over several nearby bins, as shown in Figure 51. We then iterate the frequency until this fit is maximised, and this gives an estimate of the amplitude and phase of the sinusoid. Other interpolation techniques are possible.[SmithJ 87]

[64] sinc(0) is defined as $\lim$ sinc(ε) as ε→0, which is equal to 1.

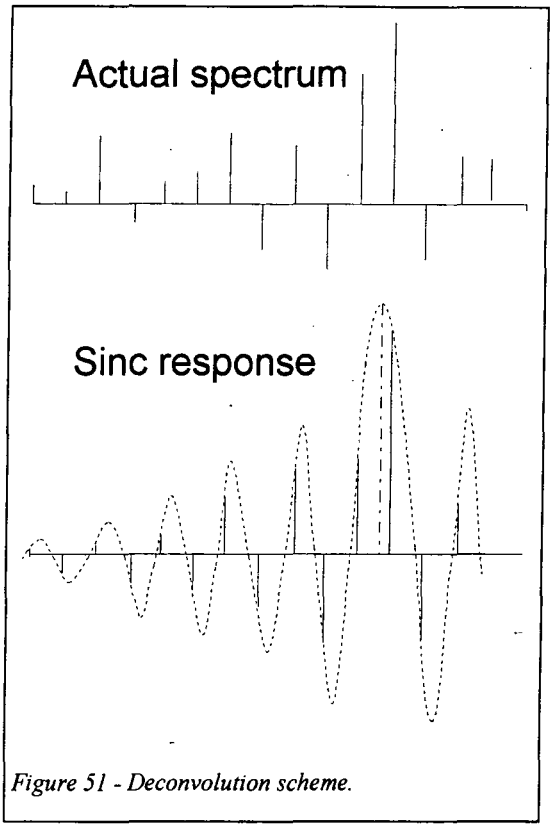


Figure 51 - Deconvolution scheme.

Note that the 'correlation' is actually between complex quantities; the 'sinc' function is also twisted around the axis, and if a window was used in the FFT, we must use the transform of the window rather than the sinc. Since we must calculate sin(x) at unknown values of x, the first value must be calculated the long way. After this, we can use recursion to calculate $\sin(x \pm nT)$. The division by x cannot be avoided – there are apparently no other shortcuts to calculating this part of the sinc function. The twist in phase of the 'sinc' function is also derived recursively.

When we have determined a sinusoid, we subtract its response from the spectrum. This is done until the residual power falls below the desired threshold, or until a given 'N' tries have failed to do so.

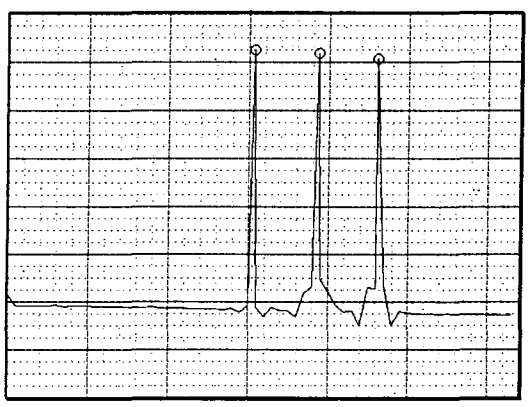
The deconvolution is by far the most time-consuming part of the process. It also has no *guaranteed* execution time – transforms with lots of energy will take longer to process.

First, a program was developed to test this deconvolution procedure. The test data was much simpler than would be likely in practice – three sharp peaks, with the lowest frequency corresponding to a position between bin 30 and bin 31 of the Fourier Transform, and two other peaks four and seven semitones higher. The data was weighted with a four-term Blackman-Harris weighting.[Harris, Nuttall 81] This has the effect of minimising the size of the sidelobes, at the expense of broadening the mainlobe. (Several other windows will be examined later to determine where the trade-off should be made.) The results are shown in Figure 52. In these graphs, a circled point indicates a value deduced by deconvolution and removed from the residual spectrum. Each vertical subdivision represents 6 dB, or 1 bit in amplitude. Each horizontal subdivision shows one of the 64 bins in the FFT.

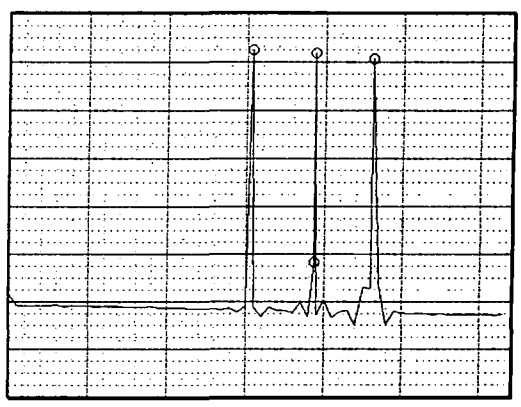
original spectrum

first peak removed

first and second peaks removed



all three peaks removed



spurious 'fourth' peak removed

Figure 52 - Stages of deconvolution.

The first diagram above shows the effect of the Blackman-Harris weighting; the peaks are much broader, but the sidelobes have been suppressed by 92 dB. Each division on the vertical axis represents 6

dB or 1 bit in amplitude.[65] After the three peaks have been removed, the residual error is down by 19 bits or 114 dB. This means that the three sinusoids will regenerate the data so accurately that we can ignore the residual. However, most typical cases will be much more complex than this (see the more typical spectrograms earlier in this section), and the peaks will be closer. Further experimentation is required before we can determine how useful this technique will be.

As there is nothing to tell the test program that this is enough, it continues to look for peaks, and finds one very close to the second peak. It would be possible to merge these two peaks into a single peak, but if we stay in the frequency domain, it will be necessary to add the response of the two peaks, and then subtract the response of the corrected peak. As the response must be removed from the entire spectrum, this will be a very computationally expensive step.

The program PICKOUT has many flags; flags for these and all other programs can be found in Appendix O. Again, the output is in the form of a large file, *.SLA (Sine List A).

6.7.3 Errors in deconvolution

The next complication arises when there are several close frequencies. In this case, the sinc responses interfere with each other. This situation is a classic test for spectral estimation techniques. There are two possible results; either the analyser will report a single strong frequency roughly halfway, or it will find the two peaks but may estimate their parameters inaccurately. When we have estimated one sinusoid, its sinc response is removed from the spectrum. This will make it easier to identify other peaks.

Naturally, errors will accumulate in the residual spectrum, so care must be taken in assuming that a small peak is actually a component of the original sound. There is, however, a possible advantage to misidentifying a spurious wriggle as a peak. For example, if the original sound contains two sinusoids, the peaks in the spectrum tend to move towards each other. If there are components at 30.3 and 31.8, then the first two frequencies to be reported might be 30.35 (higher) and 31.78 (lower). Next a small peak would be found at just below 30.3, and a fourth at just above 31.8. It would in principle be possible to spot these pairs afterwards and amalgamate them into a single component. The problem, however, is that it is not possible to represent two sinusoids of different frequencies by a single sinusoid.

6.7.4 Application to multirate analysis

The procedure described above was tested on a single FFT, but can be applied equally well to the multirate analysis system. Its effectiveness will depend on the number of sinusoids in each octave.

6.8 Filter lag compensation

The group delay of the filters in the spectral analysis was designed to be 128 samples. This is 3 ms for the top octave, but rises to 3 seconds for the eleventh. We correct for this by adjusting the times output

---

[65] An amplitude factor of 2 means a power factor of 4, and $\log_{10}(4) = 0.60206$ Bels ≈ 6 dB.

by the previous program (PICKOUT). As a result, the list of sinusoids output by the deconvolution becomes unsorted. This requires a separate sorting stage which is costly as the text output file is several megabytes in size.

The first solution was an unwieldy combination of batch files, basic programs, Borland's GREP utility[Borland] (which does not handle large files properly), and a smaller sorting program. A better solution was to write a generic large-file-sorting program called MEGASORT, which uses the virtual memory routines described below to handle the sorting procedure, and converts the *.SLA (Sine List A) file into a *.SLB (Sine List B) file.

6.9 Virtual memory

Several stages of the PC-based analysis require several MB of memory, so a simple virtual memory driver was written. The core of the handler was based on the memory allocator described in Kernighan & Ritchie.[Kernighan] It implements a virtual space of 32-bit addresses, equivalent to a maximum of 4.3 Gbytes, using, in order, near memory, far memory, extended memory[66], RAM disks[67], hard disks, compressed drives[68], and network drives.[69] The routines inform the user where each 64-kB page is being allocated, to allow him/her to monitor usage, unless `vmhush()` is called to suppress all messages. As expected, the overall speed falls as soon as the disk is required. The calling program will have its own requirements, particularly for near and far memory, so it can explicitly specify which types of memory may be used for virtual memory.

Naturally support of virtual memory adds overheads to processing, as it entails a separate function call to read and write each item in virtual memory. To alleviate this, the calling program should use buffering.

Virtual memory is used in the filter lag compensation, reordering, sine tracking, harmonic matching, and note identification stages.

---

[66] Extended (XMS) memory support uses the shareware XMSIF routines.[Birdsall] ExPAnded memory (EMS) was not present on any PC used and is not supported.

[67] The driver differentiates between RAM disks, which we would prefer to use first, and hard disks. However, RAM disks are a somewhat pointless form of virtual memory as this means using memory to simulate a disk and then using it to simulate memory. The user is notified if this situation is found.

[68] Drives compressed by DOS DBLSPACE are distinguished from conventional disks using routines supplied by Laurence Darton.[Darton] It has not been possible to determine whether this also correctly detects drives compressed by other utilities such as Stacker.

[69] DOS calls are used to get most attributes of a drive – one particular bit determines whether a drive is a network drive or not.

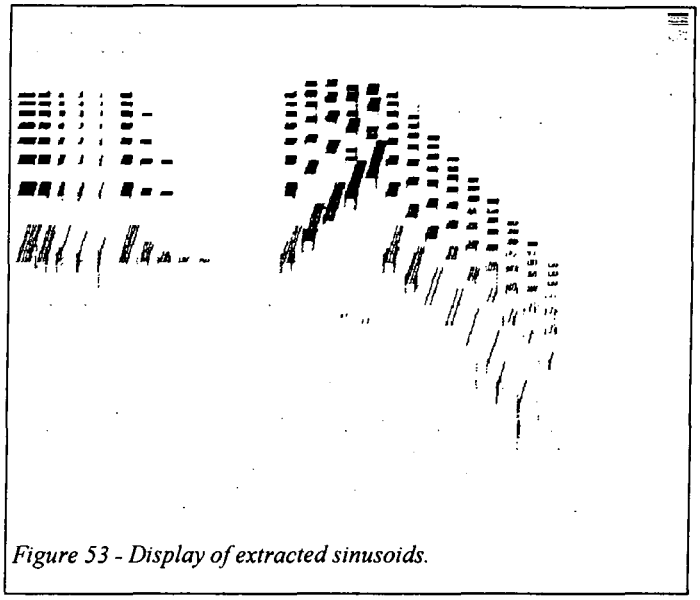


Figure 53 - Display of extracted sinusoids.

6.10 Display of extracted sines

A program called ShowSLBx was used to display the output list of sinusoids extracted. This was written in QBasic, as were several other programs where speed was not critical. A screen shot is shown in Figure 53.

6.11 Sine tracking

The sinusoids only last for between 32 and 64 periods, so we next join them by time using a program called TRAKSINx to give longer entities called chains. These are similar to the 'auditory elements' of Brown.[BrownG 94a] This is done by linking sines in neighbouring blocks using a simple birth-death model as shown in Figure 54. Note that sinusoids are only linked at their ends, so lower octaves are matched less frequently. It is possible for a sinusoid in one octave to be linked to a sinusoid in an adjacent octave.

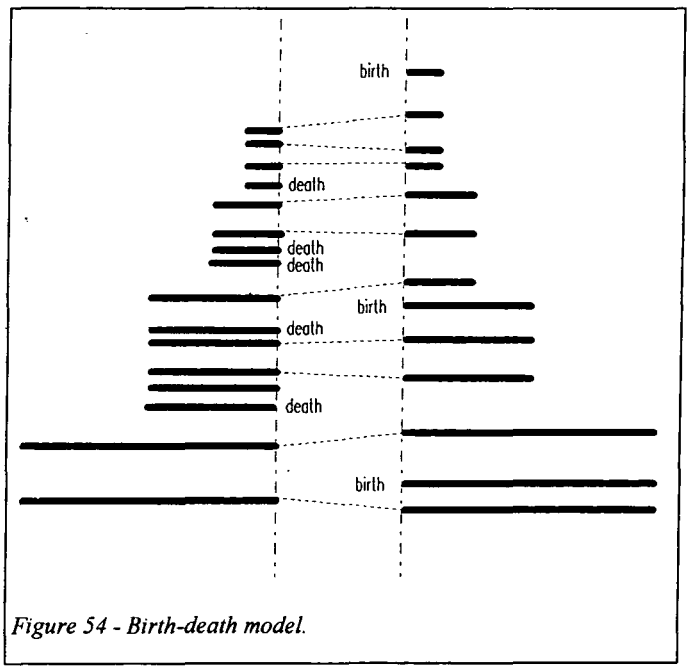


Figure 54 - Birth-death model.

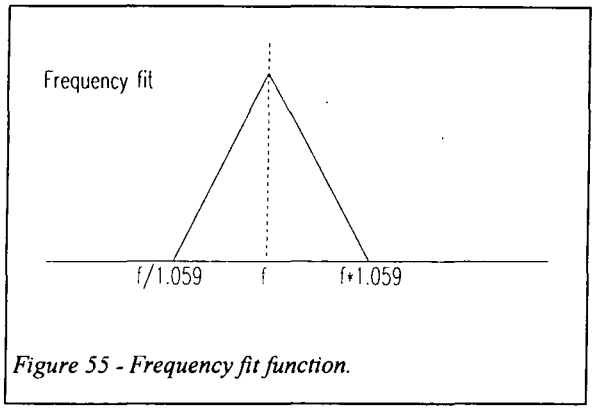


Figure 55 - Frequency fit function.

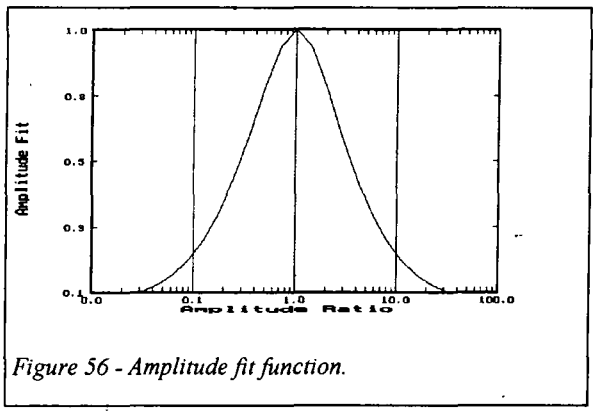


Figure 56 - Amplitude fit function.

The fit between two consecutive sines is the product of the frequency fit and the amplitude fit. The frequency fit is a triangular function of the pitch interval, falling to zero at (arbitrarily) one semitone, as shown in Figure 55. Note that this is a constant *pitch* interval rather than a constant frequency interval.

The amplitude fit function, shown in Figure 56, depends only on the ratio of the two amplitudes, and is defined as :-

$$\frac{2}{\frac{a_1}{a_2}+\frac{a_2}{a_1}}$$

We choose as many links from death to birth as possible, as long as the fit exceeds yet another threshold. The resultant sets of linked sines are referred to as chains. The list of connections between sines is written to a text file called a chain file, *.CHN.

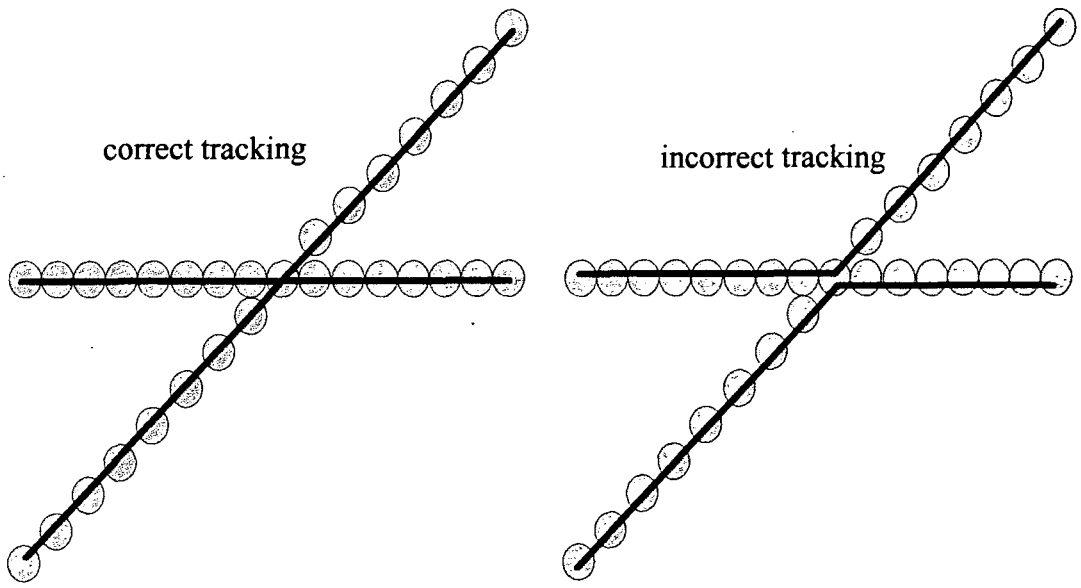


Figure 57 - Correct and incorrect partial tracking.

The birth-death model is a common choice at this stage of an analysis system.[BrownG 94a, Ellis 91] Figure 57 illustrates a problem with simple frequency tracking – when two tracks cross, there are two possible interpretations. Methods tried for solving this include Hidden Markov Models.[Depalle 93a, Depalle 93b]

6.12 Reordering

The frequency tracking outputs the chains when they are complete, i.e. when a death cannot be linked to a birth. This means that the threads are sorted in order of their end time. We wish to match harmonics from the start of the note, so it is again necessary to re-sort the data for the next process.

This program is called VREORDER. It uses the virtual memory routines described earlier. It first reads the sines into virtual memory. It then reads the chain file and outputs the tracks of linked sines (as *.REO – REOrdered list).

6.13 Track display

A QBASIC program called SHOWTRXx is used to display the output tracks. A screen shot (for MTest2) is shown in Figure 58.

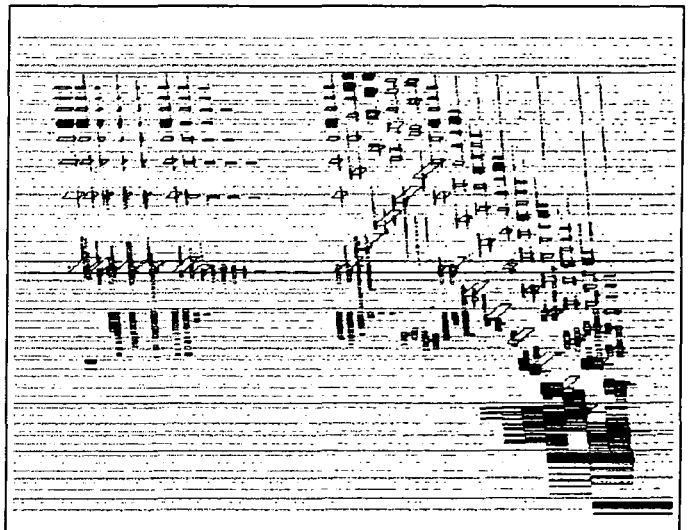


Figure 58 - Display using SHOWTRX.

6.14 Harmonic matching

The next stage is to match the partials according to the notes they belong to. This is carried out by a program called FTxxx (currently FT105).

We treat every chain as a possible fundamental of a note, and look for its harmonics, as shown in Figure 59. Since music relies on near-integer frequency ratios, we expect the harmonics of different notes to overlap. Thus, chains can be claimed several times, in which case the total amplitude is shared between all the claimants. The resultant 'note' typically contains several chains for each harmonic, overlapping in time. This technique corresponds to grouping by harmonic relations according to the 'principle of shared allocation' rather than the 'principle of exclusive allocation'.[Bregman 89]

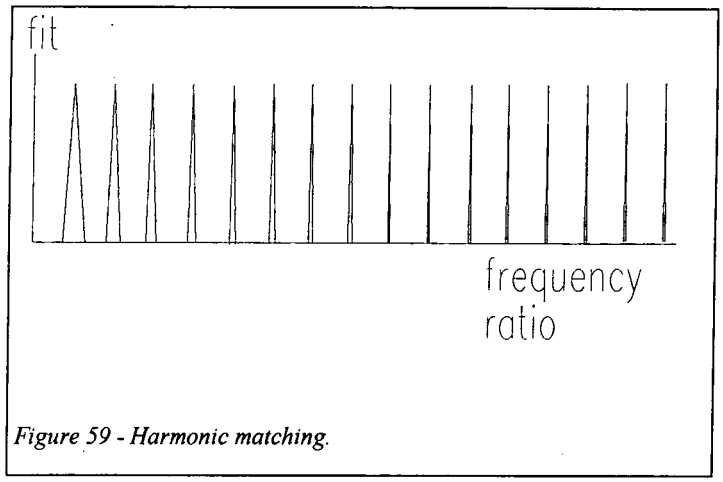


Figure 59 - Harmonic matching.

We have assumed that the fundamental is always present during the note. This, sadly, is not true, but the assumption makes processing easier. This limitation can possibly be addressed within the existing framework. We have also assumed that the note's partials are nearly harmonic. This assumption is questionable for many instruments.

First, all of the data is read into virtual memory. We then carry out an exhaustive search, looking for simultaneous frequencies that are close to being harmonically related. A partial at $(n' \times f_0)$ is taken to be the n^{th} partial of f_0 if $|n'/n|$ is less than an n^{th} of a semitone – the absolute frequency tolerance is proportional to f_0. This was chosen so that the bands do not overlap, but will not be suitable if there is considerable frequency stretching.[70]

We match up to the 16^{th} harmonic, and the resultant note groups are as shown in Figure 60. The reason that more harmonics are not sought is mainly due to memory restrictions.

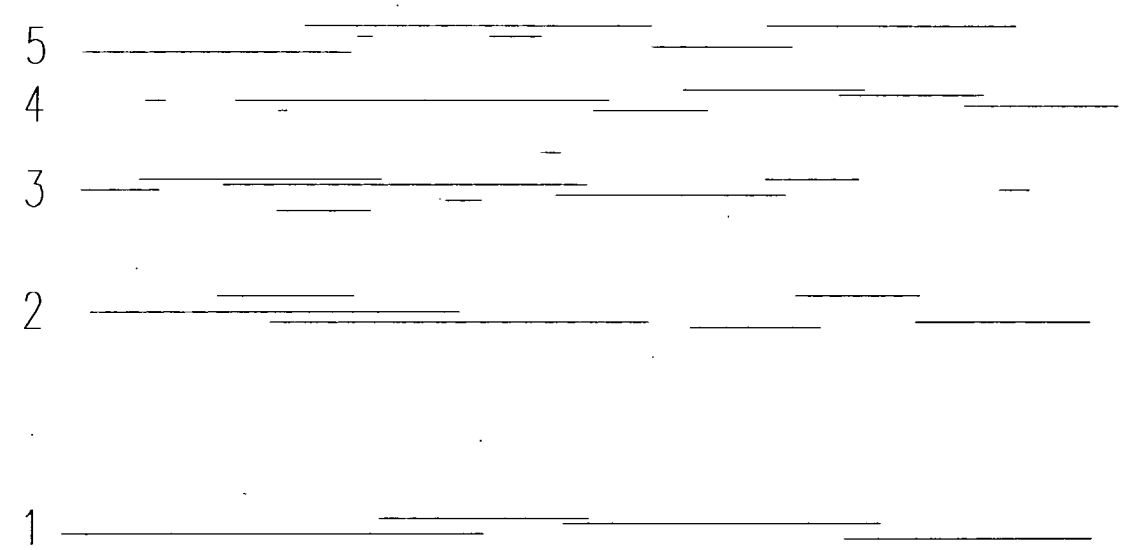


Figure 60 - Partial chains.

In the ideal case of monophonic and noise-free music, we would hope that there would be a single chain for each harmonic. However, as shown above, each harmonic typically has several chains, possibly at

[70] This contrasts with Piszczalski's matcher, where the permitted frequency error increases with the partial number.[Piszczalski 79] His system also uses much more complex weighting, whereas our is simply a pass/fail output.

two sample rates, and can have holes. We note how many times each segment of each chain is claimed and proceed to the next stage. (In practice, we dump all the virtual memory to large files on disk and the next program reloads them. As mentioned earlier, the limitations on the PC's memory prevents the whole system from being integrated seamlessly.)

6.15 Note identification

At this stage, there are far too many 'possible notes', so we must remove most of them. First, we apply several thresholds – if the note is too short or too quiet, or if we can't find many of its low harmonics, then it is killed. When we kill a note, all the sinusoids it had claimed are returned to the 'pool' for use by other notes. These rules reduce the population to a much more manageable size (typically from 14000 to 400).

We also remove notes with insufficient harmonics, and the rule stipulates that if three consecutive harmonics from the first eight are missing, the group probably arose by coincidence. This is illustrated in the table below. This 'rule' was admittedly designed with the organ spectrum in mind. Little can be said about 'typical' instrument spectra.

1	*2*	*3*	*4*	*5*	*6*	*7*	*8*	***Suitable?***
✓	✓	✓	✓	✓	✓	✗	✗	**yes**
✓	✗	✓	✗	✗	✓	✗	✗	**yes**
✓	✗	✗	✗	✓	✓	✓	✓	**no**
✓	✓	✓	✓	✓	✗	✗	✗	**no**

Table 17 - Suitability of harmonic patterns.

After this, we try to smooth the amplitude envelopes of each harmonic by adjusting the strengths of the claims. In doing this, some more notes become too weak and are killed. The processing of notes in this way is termed a 'battle', as all the potential notes are fighting each other for a share of the sinusoids found. The rules by which a note is altered or killed are under investigation; simple thresholding appears to work reasonably well for some cases such as the Mendelssohn, which has little variation in dynamics. However, it would be much more effective to adapt these thresholds to suit the incoming data.

The source signal for the Mendelssohn is played entirely on a (synthesised) organ, and many notes show the characteristic organ envelope. However, the rules still have to be extended so that we can use common characteristics of notes to derive the global instrument timbres.

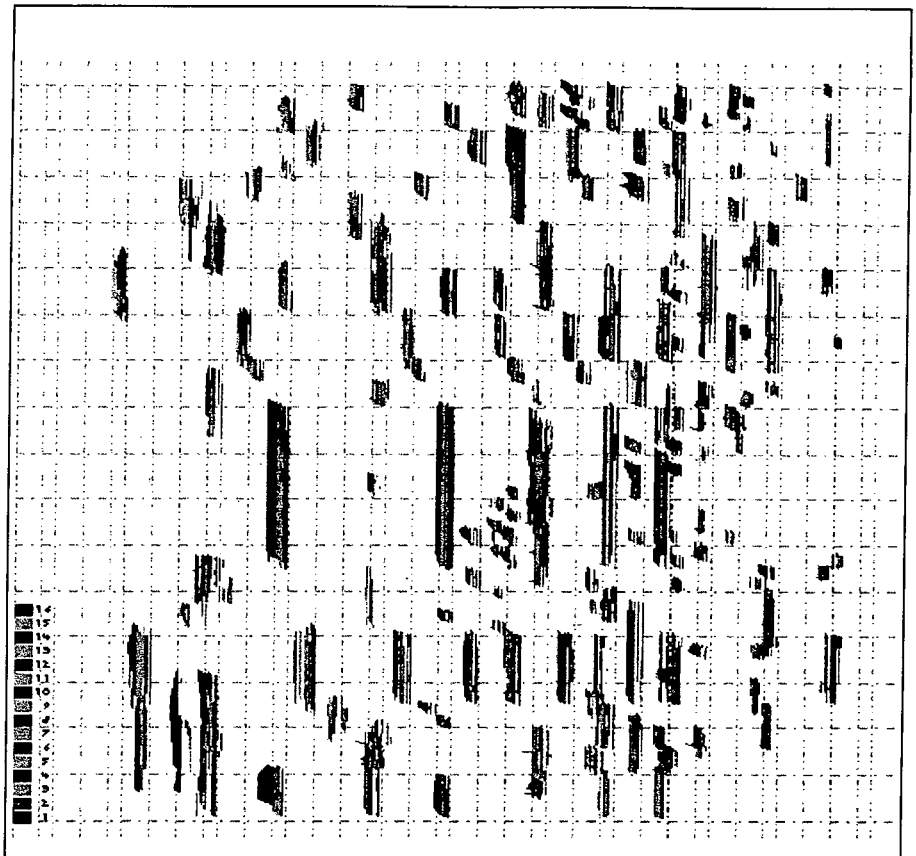


Figure 61 - Individual notes extracted from the Mendelssohn.

As each segment may be shared by many groups, the amplitude is at first shared equally between all claimants. When an unfeasible note is killed, its claims are unmade, allowing other notes to increase their claim. In order to adjust the partials to smoother envelopes, we filter the amplitude envelopes of each harmonic to get a 'better' envelope, and use the ratio to adjust the size of the claim, if possible. However, this is exceedingly complex because each partial is represented by many tracks at different frequencies.

Another rule discourages very short claims, and a third discourages frequencies further from the central frequency. We do this several (typically three) times to the remaining population, and in doing this, more notes disappear, leaving (around 220) feasible notes. This is then output as an ASCII file, which is converted using ASC2WRK and WRK2MID to a MIDI file and a Cakewalk work file, then using Cakewalk for Windows to form a common practice notation score. The graphical display given by the battle is shown in Figure 61. It shows the notes at their fundamental frequency, with the 16 envelopes derived.

6.16 Summary

I have outlined the construction of each part of the transcription system. The next chapter presents results from applying it to a variety of musical examples.

7. Transcription results

In this chapter I first outline the choices of source material to be examined. I then present results from testing the transcription system. I finish with an evaluation of the system and suggestions for future developments.

7.1 Test pieces

Many of the transcription systems described before made many assumptions about the input pieces in order to achieve robustness for a particular instrument, or for real-time performance. However, for an analysis/transformation/resynthesis system, we must design a system that can understand arbitrary musical input, at least within the framework of the definitions of music given in chapter two. It is of course practical to initially test the system using music that is reasonably 'well-behaved'.

Many of the test pieces and results are available as audio files. These are listed in Appendix Q.

7.1.1 Polyphony

Nothing has been assumed about polyphony. Defining the polyphony is difficult if several instruments play in unison. A first violin section might be described in terms of its 'source polyphony' of sixteen, or its 'note polyphony' of one.

7.1.2 Mono v stereo

The separation of sources may be easier if we can use the stereo image of the sound, particularly when the spatial locations of the instruments are fixed. Yet if we are to maintain generality, we must recognise that this is not a justifiable assumption – notes are often panned dynamically. Moreover, we still can carry out source separation on a monaural signal. Accordingly, the system is designed for a monaural signal.

7.1.3 Noise

We can use Csound[Vercoe 93] to generate a noise-free file, but a small amount of noise should not disrupt the processing. Potentially more worrying than broadband noise is the possibility of high-level transient clicks.

7.1.4 Length

The Mendelssohn test piece with which the programs were initially developed lasts for 34 seconds, and even this stretched the memory of the PC. Although a virtual memory driver was developed to allow the use of extended memory and disk-based memory, this adds an overhead. There is still a need to use standard memory for workspace, such as the pointers to whatever is being stored in virtual memory, and this was typically close to being full. Thus, the longest piece that can be handled is *probably* around 40 seconds, although this also depends on the number of sinusoids extracted and thus on the polyphony and timbres.

7.1.5 Timbre

Several assumptions are made about the timbres of the source material.

7.1.5.1 Acoustic v synthetic

We can use Csound to create mathematically precise waveforms, but real instruments will have much more complex waveforms and cannot play the same note twice.

7.1.5.2 Percussion

Most percussion instruments are unpitched broadband noise (e.g. cymbals) or have non-integral harmonics (timpani, bells, triangle). I have chosen to assume there are no such instruments in the input.

7.1.5.3 Vocals

Vowels should be handled easily, but consonants cannot yet be handled for the same reason as for percussion.

7.1.5.4 Missing fundamentals

We can clearly perceive the fundamental frequency even when there is no power at that frequency. However, this makes it difficult for the harmonic matcher to deduce which harmonics are related. (Vercoe noted problems with this case too.[Vercoe 84]) Accordingly, I have assumed that the fundamental is present throughout the note.

7.1.5.5 Inharmonicity

The algorithm allows a certain amount of deviation from exactly harmonic overtones. However, it cannot handle very inharmonic partials such as in bells and timpani.

7.1.6 Effects

Reverberation will cause notes to smear into each other, and may cause phase distortion during the note. However, reverb is a natural phenomenon that we should be able to handle robustly. Flanging and related effects also distort the waveform. In this section I assume there are no or little effects applied to the input signal.

7.1.7 Temperament

Currently, no assumptions are made about temperament and tuning. Absolute frequencies are used throughout, except for MIDI output.

7.1.8 Rhythm

The current system does not try to infer beats or bars. In some cases, particularly the Mendelssohn, the timing is artificially precise. These files are made using Csound, and as the note onsets are precisely aligned, note identification may be harder.

7.2 Results

7.2.1 Buffering experiments

The effectiveness of three tasks running on a single processor depends on how often the kernel must carry out time-slicing between the tasks. This depends on how often the buffers fill up. Thus, preliminary experiments were carried out to determine reasonable buffer sizes.

In the results given below, diskN stands for the size in words of the disk buffer. Recall that the input is 16 bits wide, so the number of samples is half of this. The bytes are sent in blocks of chanN words.[71] (There is some wastage here, as each 32-bit word is used to hold only one 8-bit byte; this is addressed later.) The analyser reads each block, converts each pair into a 16-bit integer and then to floating point, and performs the FFT.

The program was designed to allow diskN and chanN to be varied independently, subject to the proviso that one must be a power of two times the other. The input was a dummy file with a length of 1764000 or 10584000 bytes, equivalent to 10 or 60 seconds. The performance figure to be measured was the relative time, defined as the time for all the computation divided by the actual time represented by the data. For a real-time system, this figure must be below 1.

Below are the results[72], showing how the relative time depends on the sizes of these two buffers. In this experiment, the FFT was not carried out, allowing us to see the effect of the main inter-task communication. Darker shading represents a longer processing time.

Disk buffer (words)

chanN (words)	*1*	*2*	*4*	*8*	*16*	*32*	*64*	*128*	*256*	*512*	*1024*	*2048*	*4096*	*8192*
1	20.54	19.81	19.34	19.11	18.99	18.94	18.90	18.90	18.90	18.90	18.90	18.88	18.88	18.87
2	12.03	10.88	10.55	10.32	10.21	10.15	10.13	10.13	10.13	10.13	10.12	10.09	10.09	10.09
4	7.528	6.577	6.006	5.815	5.710	5.645	5.604	5.605	5.603	5.603	5.605	5.583	5.585	5.576
8	5.245	4.294	3.829	3.531	3.449	3.383	3.362	3.363	3.361	3.363	3.341	3.332	3.321	3.322
16	4.113	3.166	2.688	2.433	2.306	2.245	2.222	2.221	2.219	2.222	2.222	2.200	2.200	2.200
32	3.554	2.581	2.095	1.861	1.757	1.690	1.652	1.650	1.651	1.652	1.649	1.630	1.630	1.634
64	3.257	2.306	1.820	1.586	1.462	1.399	1.374	1.376	1.374	1.376	1.355	1.354	1.355	1.353
128	3.129	2.158	1.692	1.461	1.332	1.271	1.229	1.226	1.228	1.226	1.227	1.212	1.208	1.210
256	3.045	2.095	1.630	1.397	1.269	1.207	1.164	1.152	1.142	1.145	1.142	1.144	1.142	1.139
512	3.024	2.052	1.607	1.355	1.229	1.164	1.144	1.121	1.123	1.121	1.102	1.102	1.100	1.112
1024	3.003	2.032	1.588	1.332	1.209	1.142	1.123	1.100	1.102	1.099	1.101	1.099	1.095	1.089
2048	3.001	2.031	1.567	1.332	1.207	1.142	1.123	1.100	1.100	1.079	1.081	1.079	1.081	1.079
4096	2.982	2.031	1.567	1.334	1.205	1.144	1.103	1.099	1.081	1.079	1.081	1.079	1.081	1.079
8192	2.984	2.029	1.566	1.327	1.205	1.144	1.101	1.092	1.079	1.079	1.081	1.079	1.081	1.079

Table 18 - Dependence of OSA timing on disk and FFT buffer sizes.

---

[71] On the C40, the smallest addressable unit of memory is a word, which is 32 bits wide.

[72] These results are from version 1.101 of the OSA routines. As the software is constantly being improved, it will be of little use to directly compare these figures with later results.

This data can be more easily interpreted as the graph in Figure 62. Intuitively, increasing the sizes of buffers increases the performance, but the performance approaches a constant fairly quickly. It is also clear that it is more important to increase the size of the channel buffer than the disk buffer. Note also that real-time performance marginally not possible, but could possibly be achieved using shorter filters.

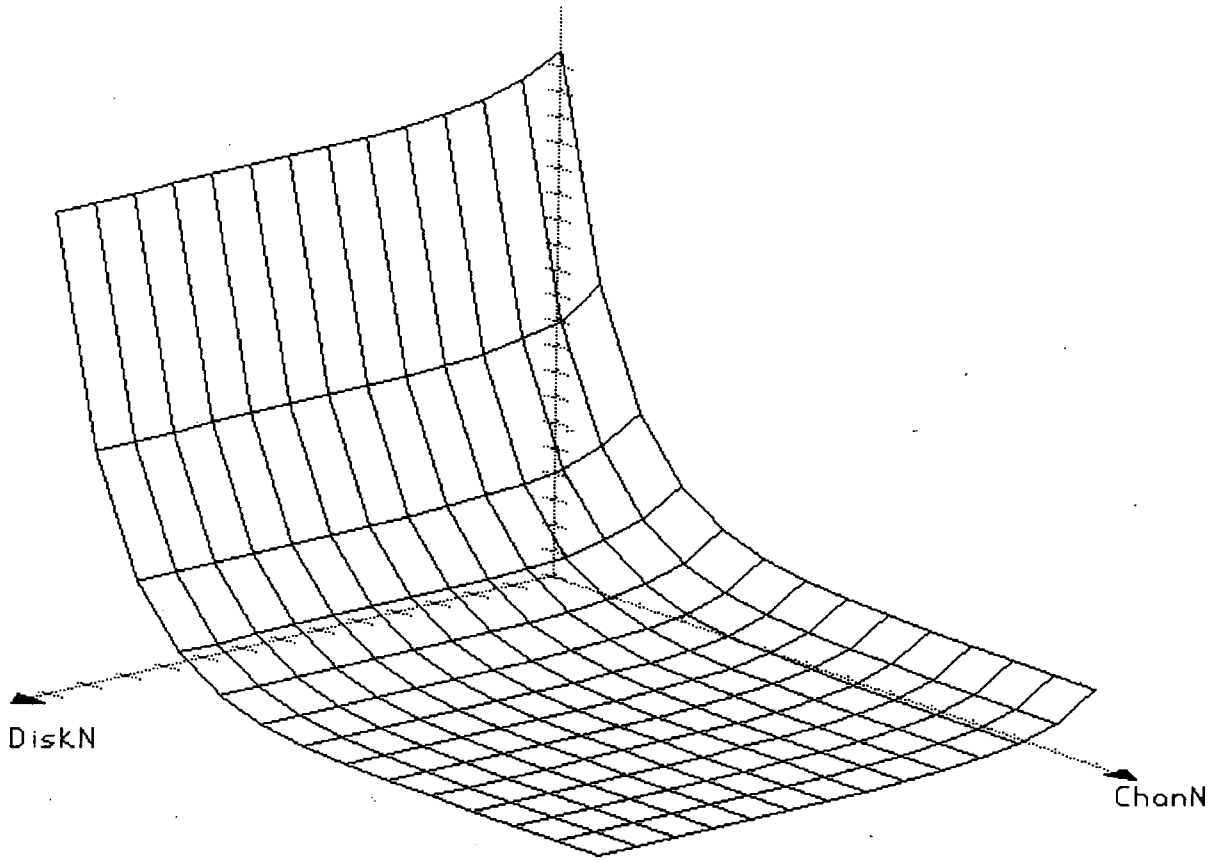


Figure 62 - Timing of inter-task communication.

7.2.2 MTest1 and MTest2

Two monophonic test pieces, with the same score but different timbres, were written to confirm the workings of the analysis system. These form (((audio examples 1 and 2))) The pieces were around half a minute long and were made at 16000 Hz by Csound.[Vercoe 90, Vercoe 93] The Csound control rate is often set to a figure of around 20 ms, but this can cause distortion. To prevent this, the control rate was set equal to the sample rate. The resultant synthesis may be 7 times slower.[Dannenberg 92]

7.2.2.1 MTest score

The pieces have several groups of notes of varying length, volume, and frequency, in order to highlight any deficiencies in the method. The score is listed in the table below and illustrated in Figure 63.

start (s)	duration (s)	periods	amplitude	Csound note	frequency	MIDI note
0	0.8	352	8192	8.09	440	A 4
1	0.5	220	"	"	"	"
2	0.2	88	"	"	"	"
3	0.1	44	"	"	"	"
4	0.05	22	"	"	"	"
5	0.5	220	8192	8.09	440	A 4
6	"	"	4096	"	"	"
7	"	"	2048	"	"	"
8	"	"	1024	"	"	"
9	"	"	512	"	"	"
10	"	"	256	"	"	"
11	"	"	128	"	"	"
12	"	"	64	"	"	"
13	0.5	220	8192	8.09	440	A 4
14	"	311.1	"	9.03	622.2	Eb 5
15	"	440	"	9.09	880	A 5
16	"	622.2	"	10.03	1244.5	Eb 6
17	"	880	"	10.09	1760	A 6
18	0.5	220	8192	8.09	440	A 4
19	"	155.6	"	8.03	311.1	Eb 4
20	"	110	"	7.09	220	A 3
21	"	77.8	"	7.03	155.6	Eb 3
22	"	55	"	6.09	110	A 2
23	"	38.9	"	6.03	77.8	Eb 2
24	"	27.5	"	5.09	55	A 1
25	"	19.4	"	5.03	38.9	Eb 1
26	"	13.8	"	4.09	27.5	A 0

Table 19 - Score of MTest1 and MTest2.

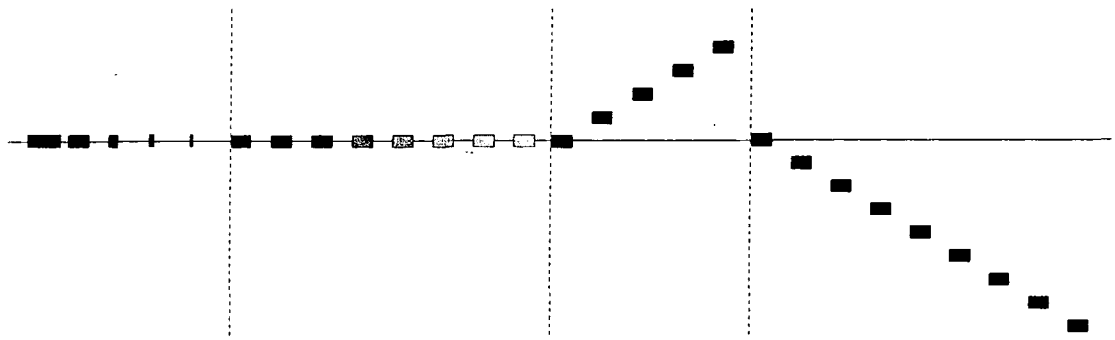


Figure 63 - Schematic of score of MTest1 and MTest2.

7.2.2.2 Test timbres

Two timbres were investigated. MTest1 uses a sine wave, and MTest2 uses a band-limited square wave, shown in Figure 64. The latter had odd harmonics 1-15,[73] and caused aliasing for notes 14-17. This can be seen in Figure 65 below and heard in the audio example.

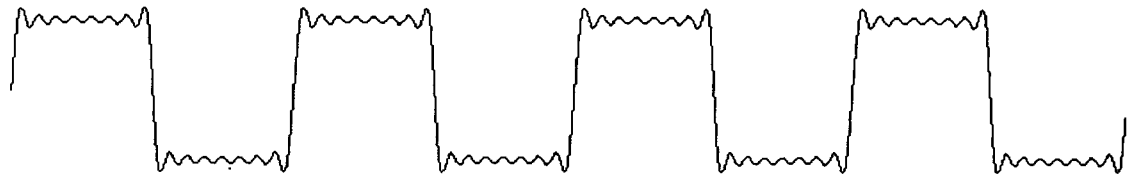


Figure 64 - Waveform of MTest2.

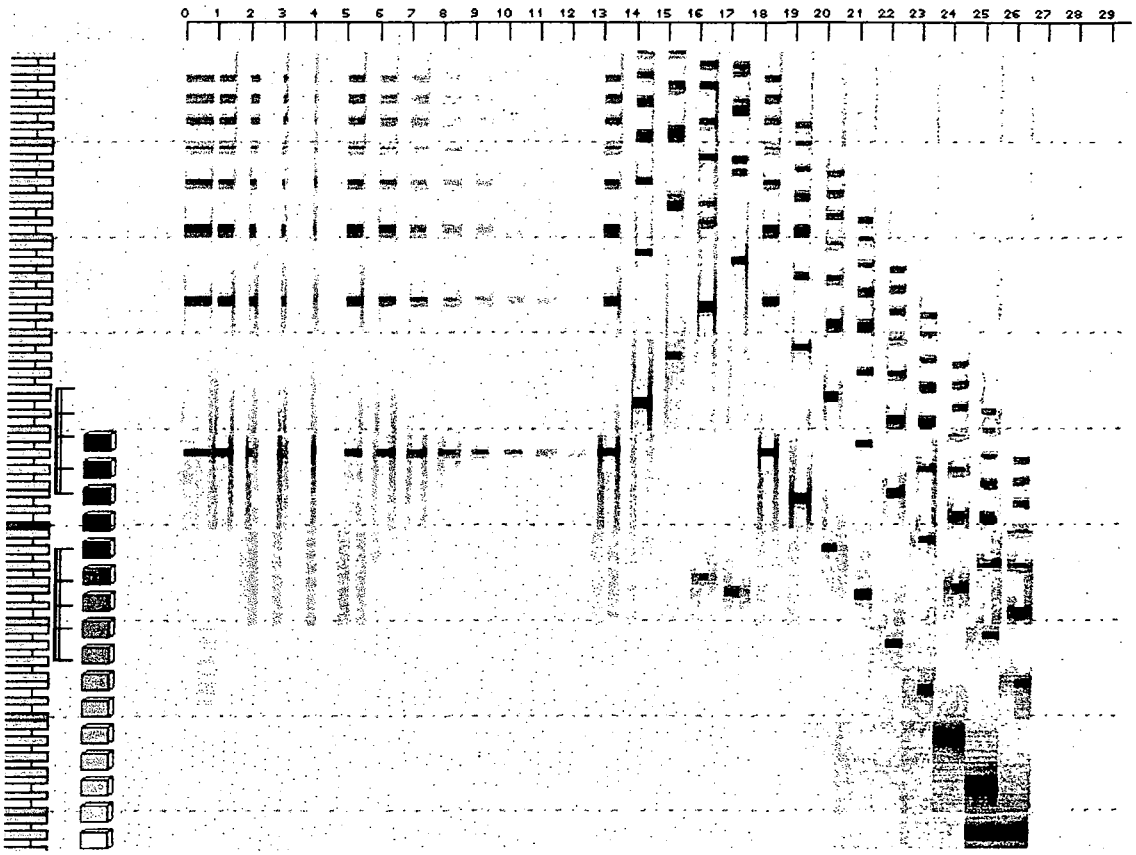


Figure 65 - Spectrum of MTest2.

7.2.2.3 FFT size

Another early choice is that of the FFT size. This must be a power of 2, and determines the trade-off between time resolution and frequency resolution, as illustrated in the table below.

---

[73] MTest2.Sco defines the timbre using the line:-

```
f1 0 4096 10 1 0 0.333333 0 0.2 0 0.142857 0 0.111111 0 0.090909 0 0.0769231 0 0.0666666
```

FFT size N	av. Δf (semitones)	Δt (s) f=7040 Hz (a9)	Δt (s) f=1760 Hz (a7)	Δt (s) f=440 Hz (a5)	Δt (s) f=110 Hz (a3)	Δt (s) f=27½ Hz (a1)
256	0.09	.032	.128	.512	2.048	8.192
128	0.19	.016	.064	.256	1.024	4.096
64 (default)	**0.38**	**.008**	**.032**	**.128**	**.512**	**2.048**
32	0.75	.004	.016	.064	.256	1.024
16	1.5	.002	.008	.032	.128	.512

Table 20 - Dependence of bandwidths on FFT size.

Earlier tests had examined FFT sizes of 64, 128, and 256, and had suggested that N=64 gave the best trade-off between time and frequency resolution. This was done by visually comparing the Mendelssohn spectra, which had no very short notes. However, the lowest quavers in the bass line had 13 periods, and thus were still too small for the time resolution. It appears that 64 may still be too large, as the mid-range resolution (440 Hz) is 15.6 blocks per second. Most of the analyses use either 64 or 32. For this piece, we will continue assuming an FFT size of 64.

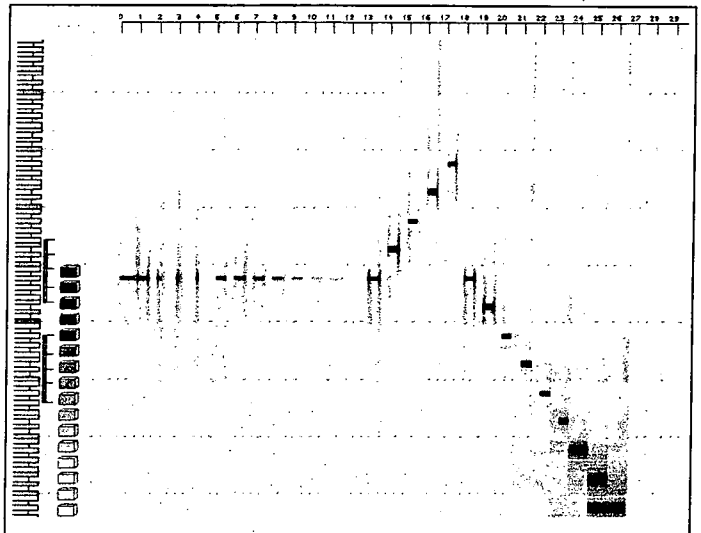


Figure 66 - Spectrum of MTest1 with FFT size of 64.

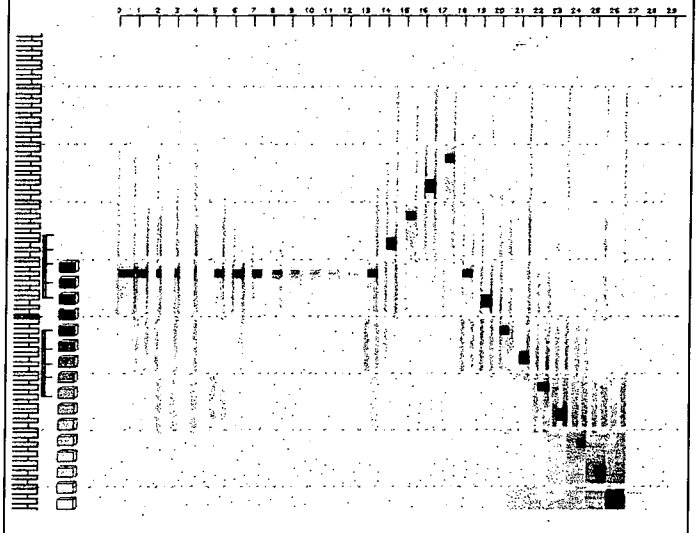


Figure 67 - Spectrum of MTest1 with FFT size of 32.

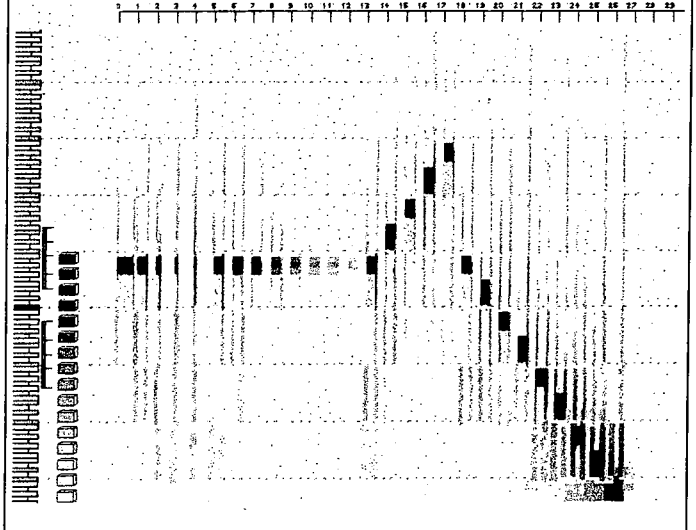


Figure 68 - Spectrum of MTest1 with FFT size of 16.

Figure 66, Figure 67, and Figure 68 show the spectra of MTest1 with FFT sizes of 64, 32, and 16 respectively. (The picture with an FFT size of 32 shows a dead spot in the spectrum at each frequency where two octaves meet - this is the result of an earlier bug in the multirate analysis.) In these figures, the amplitudes are only plotted using the greyscale, rather than the 'skyscraper' effect shown earlier. With an FFT of size 64, the frequency resolution is better (an *average* of 3/8 of a semitone or 37½ cents), but the time resolution is poorer. This can be seen most clearly for the last few low notes. With smaller FFT sizes, the time resolution is better but the frequency resolution is worse (¾ of a semitone for N=32 and 1½ semitones for N=16).

7.2.2.4 Characterisation

Distrib2 showed that MTest1 has an average level of -16.94 dB and MTest2 has an average level of -16.49 dB.

7.2.2.5 Windowing

The notes have rectangular envelopes. The rapid onsets and offsets cause a significant amount of energy in the sidelobes, regardless of the FFT size. This gives many notes a characteristic 'H' shape unless the start happens to coincide with the start of a block. This would appear to be an error but it is correct – given

that we have asked the FFT to describe what is found in this block, it has 'correctly' told us that the block contains high frequencies. This is due to the fundamental assumption of the FFT that the block it analyses is periodic, as illustrated in Figure 69.

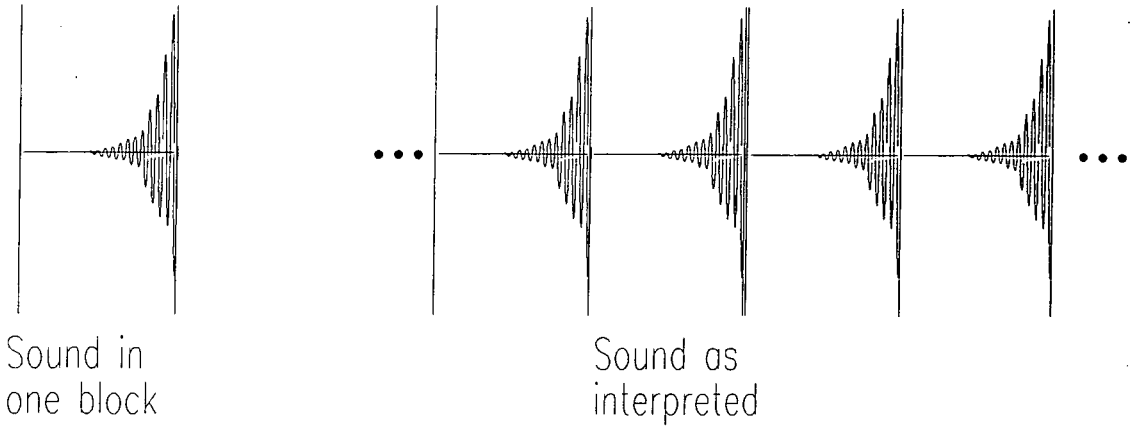


Figure 69 - Spectral distortion caused by blocks.

This problem can be alleviated by the use of windows. Several windows were available, and are shown for MTest1 in Figure 70 to Figure 73 below.[Nuttall 81] Although they are shown using a greyscale, these figures were originally made using the 8-colour colour scheme, as shown by the key at the left. The windows, apart from the triangular window, are of the form:-

$$w(i)=a_0-a_1*\cos(2\pi*i/N)+a_2*\cos(2*2\pi*i/N)-a_3*\cos(3*2\pi*i/N)$$

and the values are given in the following table.

Window	*Type*	a_0	a_1	a_2	a_3
0	none	1			
1	Hamming	0.54	0.46		
2	Hann	0.5	0.5		
3	Triangular	n/a	n/a	n/a	n/a
4	Blackman 2-term	0.42	0.5	0.08	
5	Blackman exact 2-term	7938/18608	9240/18608	1430/18608	
6	Blackman 3-term	0.44959	0.49364	0.05677	
7	Blackman minimum 3-term	0.42323	0.49755	0.07922	
8	Blackman 4-term	0.40217	0.49703	0.09892	0.00188
9	Blackman minimum 4-term	0.35875	0.48829	0.14128	0.01168

Table 21 - Parameters of windows used in analysis.

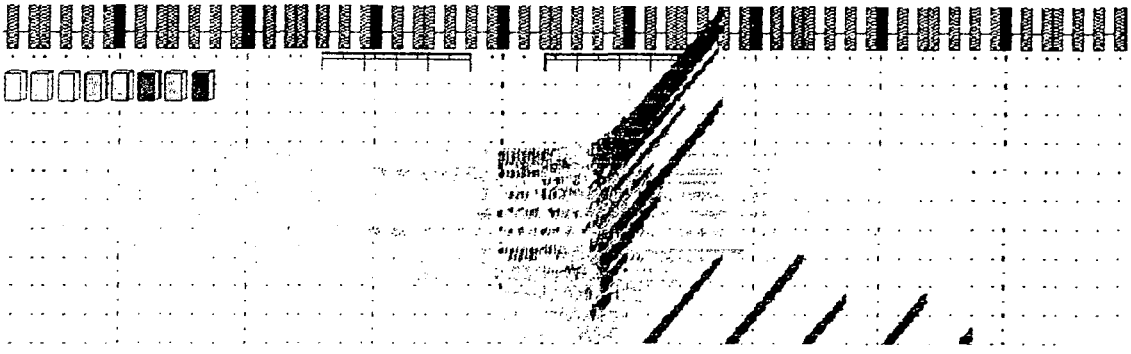


Figure 70 - No window.

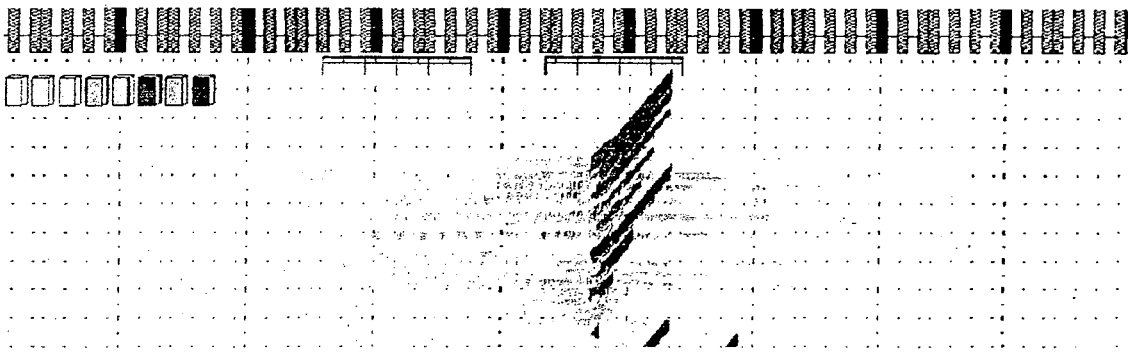


Figure 71 - Hamming window.

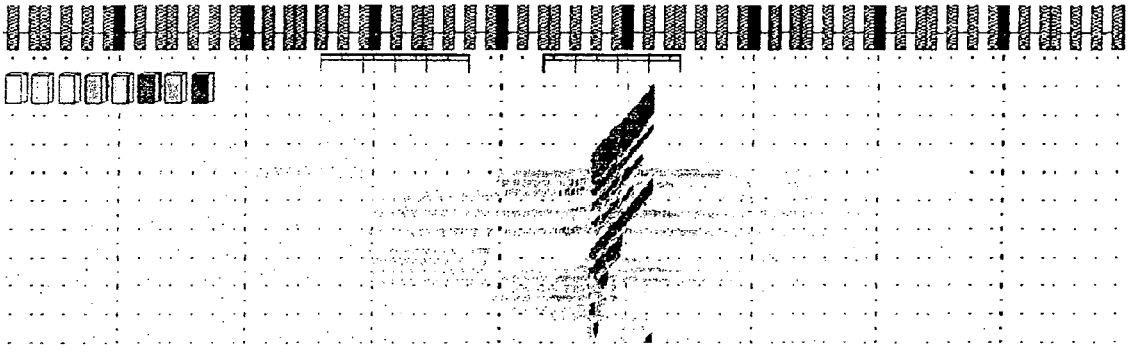


Figure 72 - Blackman 2-term window.

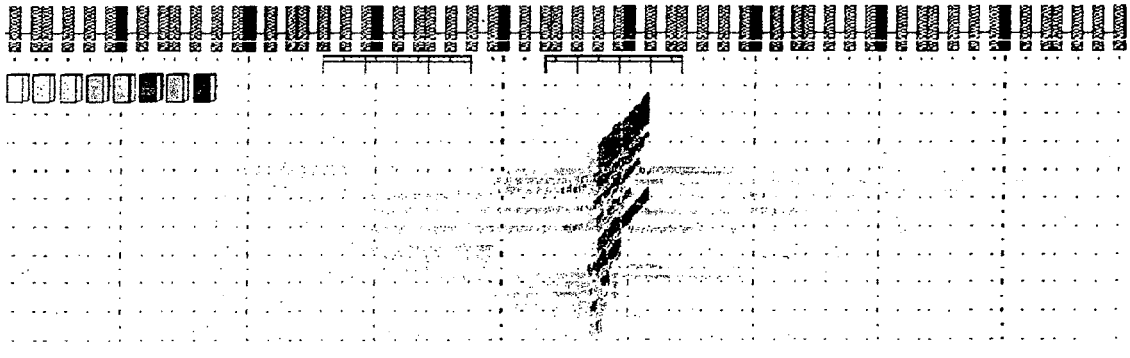


Figure 73 - Blackman minimum 4-term window.

It can be seen that any window is better than no window. However, more complex windows mean a more complex deconvolution process. For this reason, all of the following analyses used the Hamming window.

7.2.2.6 Duration and time resolution

The first five notes examined the effect of changing the duration of the note while the frequency and amplitude are fixed. The critical figure is the time resolution, which was earlier determined to be 64 ms in this octave. For a 440-Hz note, the period is 2.2727 ms. This represents 400/11 = 36.3636 samples. The total durations of the notes are shown below.

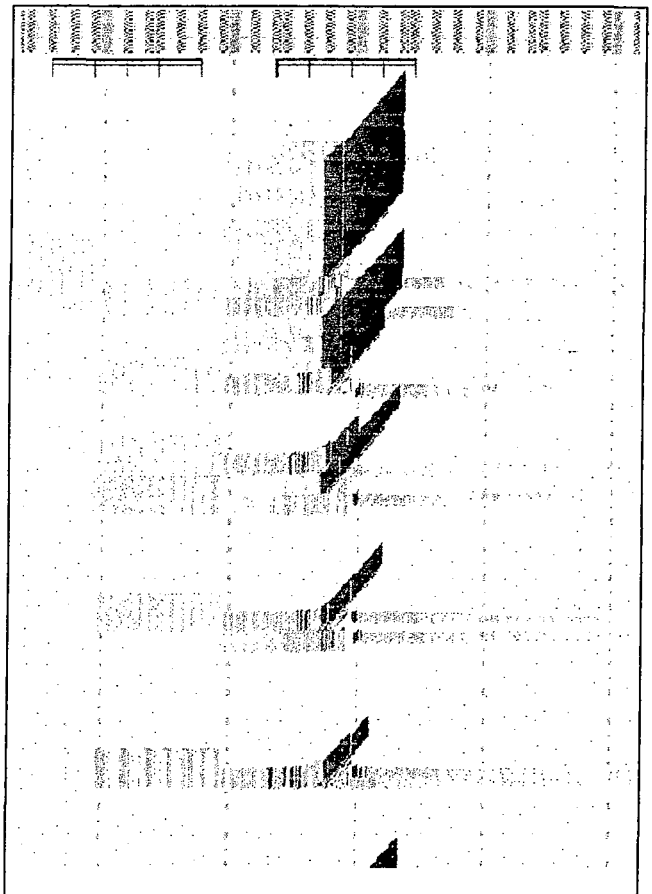


Figure 74 - Weakening of short notes.

duration	samples	periods
0.8	12800	352
0.5	8000	220
0.2	3200	88
0.1	1600	44
0.05	800	22

Table 22 - Samples and periods in the first 5 notes of MTest1 and MTest2.

We can anticipate problems in detecting the shortest of these notes. In the spectral analysis, the FFT size normally used was 64. This means that the block would represent between 32 and 64 periods of a frequency (depending on its position within that octave). The shortest note would only be present for a part of one or two blocks, and would appear to be a proportionately weaker component. We could expect this to affect the third, fourth, and fifth notes. The spectra, shown in Figure 74, show that this is indeed the case. The first two notes are long enough for their maximum amplitudes to be correct. The third is very slightly lower, showing the onset of the weakening effect. It has 88 periods, and thus falls into either two or three blocks. A close examination of the spectrum shows that the former is the case, as illustrated in Figure 75. The fourth and fifth notes show the weakening effect more clearly. Their amplitudes are around 77% and 59% of the actual amplitude.

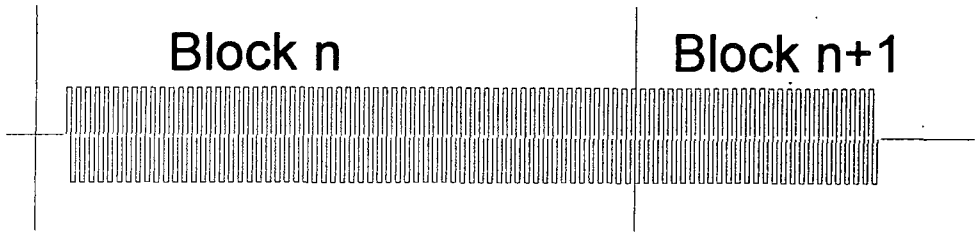


Figure 75 - Weakening effect of partially-filled blocks.

This effect can also be observed in the last nine notes as the frequency falls in steps of half an octave, as shown in the table below. All of these have a duration of 0.5 s. Again, even the shortest sine may be included in 2 blocks if it falls over a boundary.

frequency	*note*	*periods*	*approx. blocks*
440	a5	220	5
311.1	eb5	155.6	4
220	a4	110	2
155.6	eb4	77.8	2
110	a3	55	1-2
77.8	eb2	38.9	1-2
55	a2	27.5	1-2
38.9	eb1	19.4	1-2
27.5	a1	13.8	1-2

Table 23 - Periods and blocks in lowest notes of MTest1 and MTest2.

At this stage, it is informative to consider what are the shortest and lowest notes that are likely to occur. The normal range of bass instruments ends at around 40 Hz. The lowest pitches on several contrabass instruments were discussed in an earlier chapter, with the conclusion that the lower end of the bass range was *normally* at 30 Hz and *exceptionally* at 15 Hz. A typical 'very short' note would be a 40-Hz semiquaver at 120 bpm. This would be equivalent to 1/8 s, so there would only be 5 periods of the wave. However, this only applies to the fundamental; most bass instruments have little energy at such low levels.

7.2.2.7 Deconvolution and sine tracking

The deconvolution sets a threshold for the allowable amount of power left in the spectrum. This is denoted in decibels below the average level – which is calculated in an earlier stage. To show how varying this threshold affects the performance of the deconvolution program PICKOUT, eight values were examined:- -6, -12, -18, -24, -30, -36, -42, and -48 dB.

MTest1 should have around 137 sine waves, according to ExpNoSin.Bas, and 27 partials. The table below shows how many were picked out, and how many partial tracks were formed; the data is illustrated in Figure 76.

Threshold	*Sines extracted*	*Partial tracks*
-6	97	26
-12	151	75
-18	355	273
-24	669	519
-30	1114	908
-36	2039	1730
-42	4404	2990
-48	5914	3987

Table 24 - Summary of analysis of MTest1.

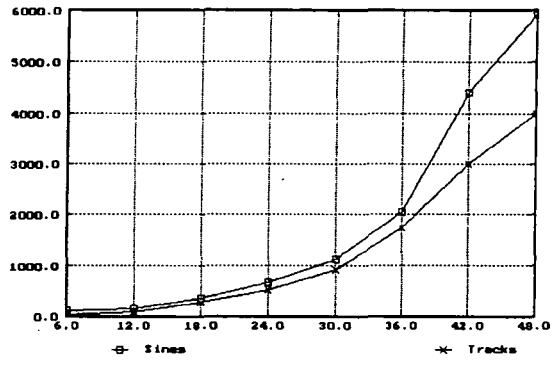


Figure 76 - Sines and partials for MTest1.

The results of the deconvolution for MTest2, the square-wave piece, are plotted using ShowSLBx.Bas, and are shown in the eight pictures in Figure 77. Mention was made in the previous chapter of the programs used to animate the output of the spectral display program. This can also be applied to other sets of images such as those below. This produces an animation of 'the effect of altering the deconvolution threshold on the sines extracted'.

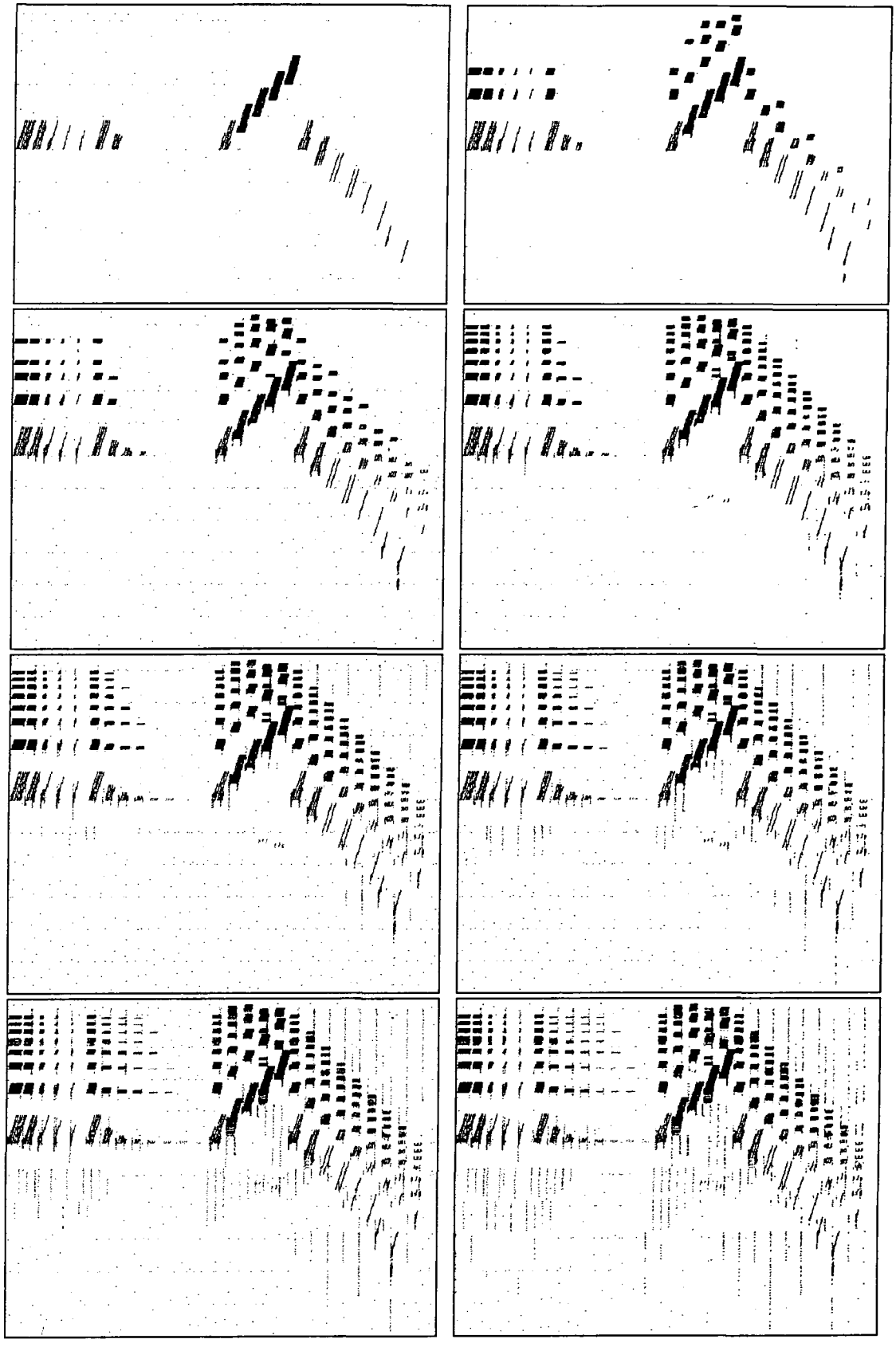


Figure 77 - Results of deconvolution for eight thresholds.

It can be seen that many spurious sinusoids are being picked out as the threshold decreases. The numbers of sines removed from MTest2 is shown in a later table. The actual number of sines that should be removed is determined by another QBasic program (ExpNoSi2.Bas), and the 'correct' number for MTest2 is 7618. This would seem to indicate that -30 dB is an appropriate threshold.The sinusoids are then tracked using the procedure described in the previous chapter. Below are the outputs plotted by SHOWTRX for MTest2 for the eight deconvolution thresholds.

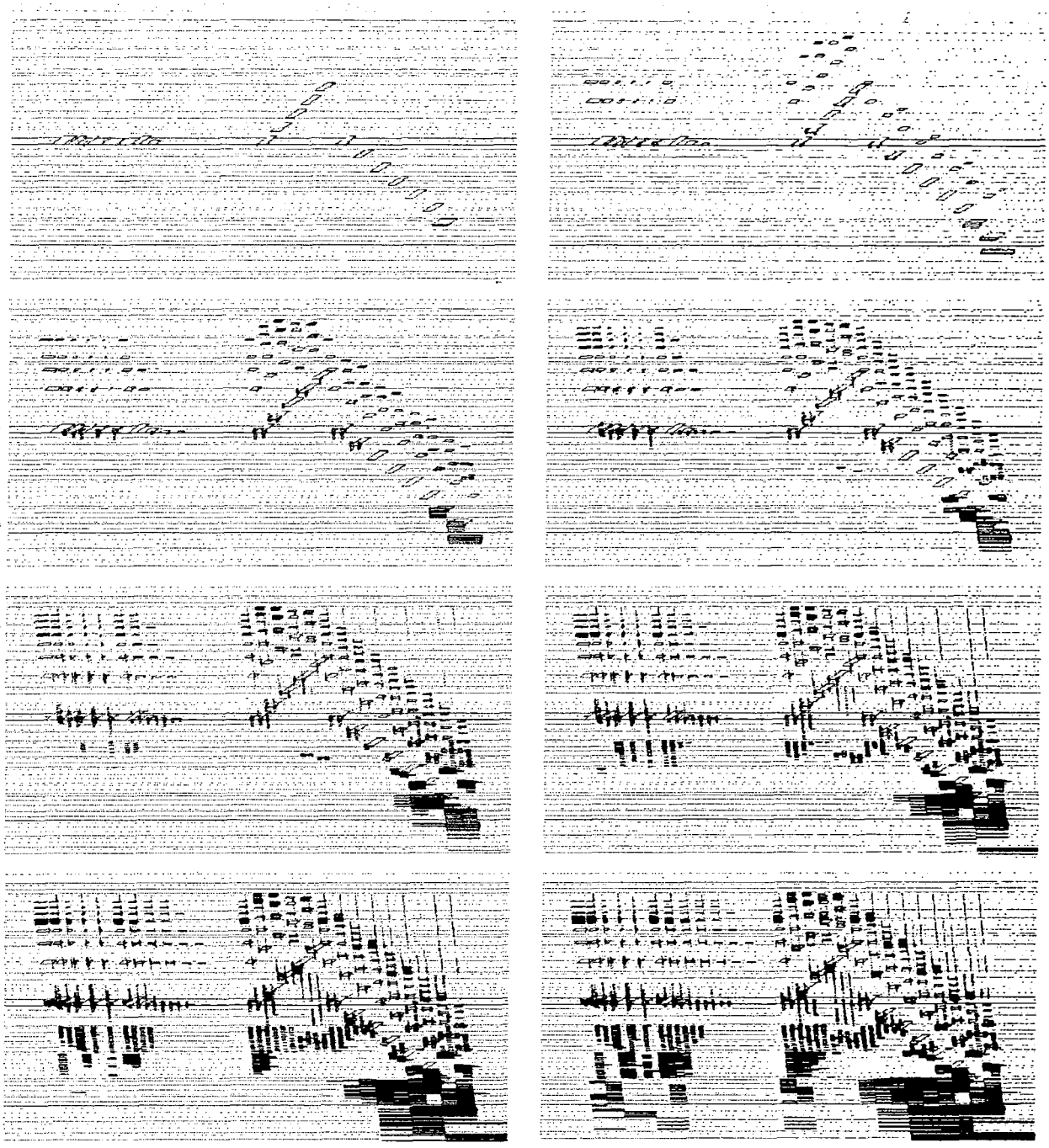


Figure 78 - Tracked partials for eight thresholds.

The number of partial tracks is shown in the table below and in Figure 79. The correct numbers are 7618 sines and 405 tracks.

Threshold	*Sines extracted*	*Partial tracks*
-6	99	26
-12	894	163
-18	2435	418
-24	5138	1078
-30	7624	2684
-36	11678	4735
-42	18168	6802
-48	27585	9642

Table 25 - Summary of analysis of MTest2.

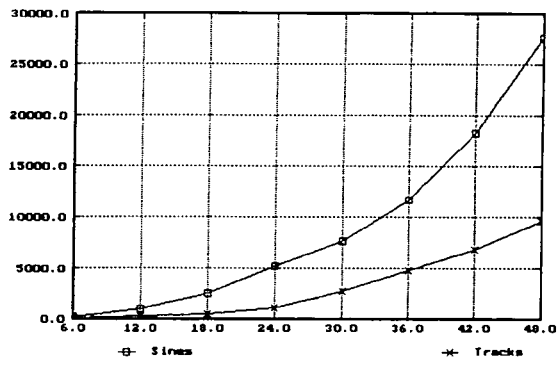


Figure 79 - Sines and partials for MTest2.

7.2.2.8 Recognition of quiet notes

After the first five notes, the next eight have different amplitudes. The amplitudes are 8192, 4096, 2048, 1024, 512, 256, 128, and 64. The table below shows which of the harmonics of the wave were recognised. The values of the threshold are shown along the top, and the values at the side show the amplitude of the fundamental. The results are well organised, due to the fact that a change in threshold of 6 bits is equivalent to a halving of amplitude.

	-6	-12	-18	-24	-30	-36	-42	-48
8192	1	1-5	1-11 *	**1-15**	**1-15**	**1-15**	**1-15**	**1-15**
4096	1	1	1-5	1-11 *	**1-15**	**1-15**	**1-15**	**1-15**
2048	-	1	1	1-5	1-11 *	**1-15**	**1-15**	**1-15**
1024	-	-	1	1	1-5	1-11 *	**1-15**	**1-15**
512	-	-	-	1	1	1-5	1-11 *	**1-15**
256	-	-	-	-	1	1	1-5	1-11 *
128	-	-	-	-	-	1	1	1-5
64	-	-	-	-	-	-	1	1

* 9th harmonic missing

Table 26 - Harmonics of quiet notes recognised.

The deconvolution threshold has a predictable effect – lowering the threshold allows us to extract more sinusoids. Curiously, in the cases marked * above, the 9th harmonic is not recognised even when the weaker 11th is. This can be explained by noting that the 9th harmonic is at 3960 Hz. This would be at the edge of the 2-4 kHz octave and would be attenuated due to the non-ideality of the half-band filter. Although the attenuation is small, it is clearly enough to drop the sine below the recognition threshold.

7.2.2.9 Harmonic matching and MIDI output

Next we turn to the harmonic matching stage. MTest1, having a sine timbre, should not be transcribed at all. We will therefore examine MTest2.

It is analysed by Battle (v0.28) with a minimum time for notes of 0.05 seconds (and MINPOWER=200). The table below shows the results. The column 'trivial' shows potential notes that were found to only have ONE partial – i.e. the fundamental.

deconvolution threshold (dB)	*potential notes*	*trivial*	*too short*	*bad harmonics*	*too quiet*	*notes left*
-6	26	19	0	7	–	0
-12	162	98	13	52	–	0
-18	418	108	87	210	5	8
-24	1078	206	364	455	31	22
-30	2684	277	1454	787	127	39
-36	4735	262	3028	1062	323	60
-42	6802	223	4309	1635	572	63
-48	9642	412	6204	2124	833	69

Table 27 - Notes removed and remaining after battle.

The original score and the derived scores are shown in Figure 80. This diagram (for the -24 dB threshold) was produced by a program called READASC, which plots two scores in 'piano-roll' notation. Notes in the original are shown in light grey, notes derived are shown in mid grey, and where these coincide, corresponding to 'correct' transcription, is shown in dark grey – this corresponds to the holes we would see if we held two piano rolls together. It is noted that this assumes a quantisation of frequency to a twelve-tone set and would be unsuitable for comparing notes with glissandi.

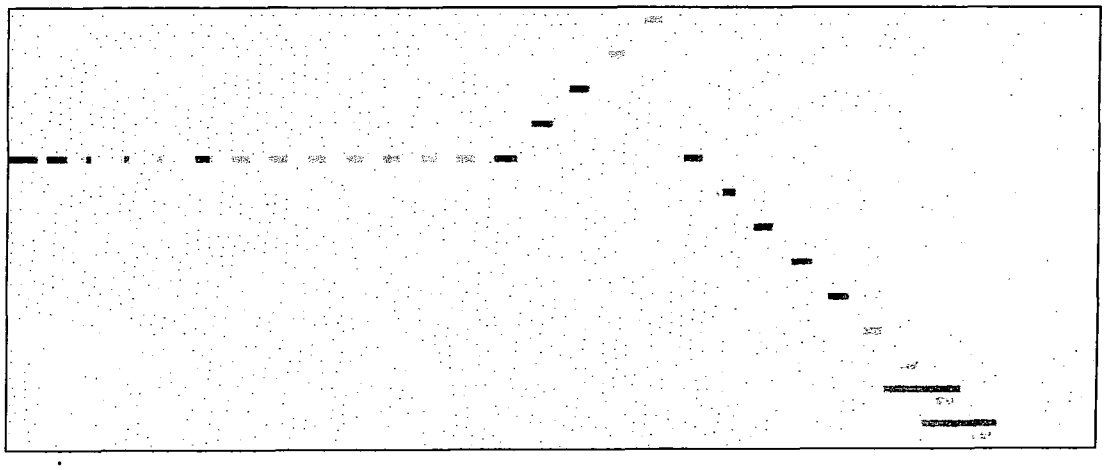


Figure 80 - Display for score comparison.

Figure 81 shows the outputs for all eight thresholds – -6, -12, -18, -24, -30, -36, -42, and -48 dB. It is clear that with a lower threshold more possible notes are removed, but many are still removed for being

unfeasible. The most obvious error for lower thresholds is the presence of many long low notes, where a single block is enough for low notes to exceed the minimum time threshold. If some energy above falls close to some harmonics, then it gives the spurious low note sufficient harmonics to appear reasonable. We might think that the 'bad harmonics' criterion should apply. However, a note is taken to possess a harmonic if the harmonic is present *at any time for any length* during the fundamental. This means that a long fundamental can claim any short frequencies, and the fleeting presence of partials 1, 3, and 6 is sufficient to suggest a valid note.

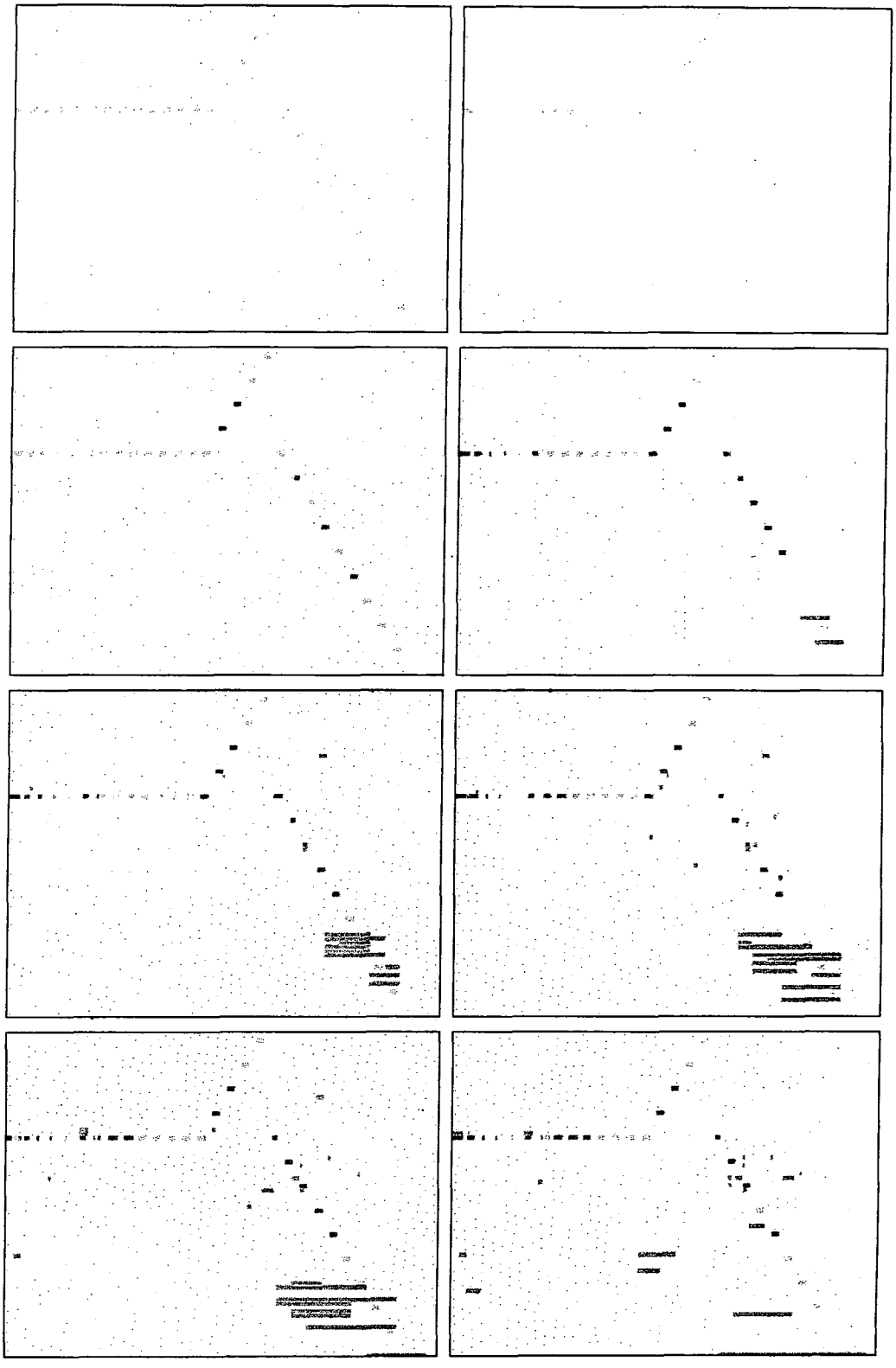


Figure 81 - Comparison of original and derived scores for MTest2.

In order to be able to quantitatively judge the accuracy, a scheme based on the above diagrams was devised. At each point in time, there are two sets of notes – those in the correct score, and those in the

output score. Thus we 'flatten' the above display by summing areas in the above figure to get the diagram shown in Figure 82. This is also produced by ReadAsc.

Figure 82 - Global comparison of scores for MTest2.

This is interpreted in the same way – light grey means missed notes (false negatives), mid grey means added notes (false positives), and dark grey means a correct identification.

It is useful to have a single figure to represent the accuracy, and this relates to how strongly the sets of notes overlap. We wish to punish both notes that have been missed and extraneous notes. Thus we define our accuracy as the *area* of overlap divided by the total shaded area, or (correct polyphony)/(correct + missed + added). Note that by using the area in this way, we do not place undue penalty on getting the start or end time of a note wrong; this simply counts as a slightly reduced area of overlap. There is also little penalty in transcribing a semibreve as four legato crotchets; the score depends on the total duration of overlap alone, and the comparison does not try to ascribe one note in the transcription to one note in the real score. The 'overlap' and 'total' columns in the table can be misleading, as will be seen later.

The outputs for MTest2 can be summarised in the following table. It shows the total number of notes in the transcription, in the ideal score, in the overlap, and the total. It also gives the average polyphony in these categories[74], and these are used to calculate the final 'accuracy' figure.

	NOTES				POLYPHONY				
threshold	guessed	ideal	overlap	total	guessed	ideal	overlap	total	accuracy
-6	0	27	0	27	0	0.385	0	0.385	0
-12	0	27	0	27	0	0.385	0	0.385	0
-18	8	27	7	28	0.113	0.385	0.076	0.422	0.181
-24	22	27	18	31	0.436	0.383	0.184	0.634	0.291
-30	39	27	19	47	1.588	0.383	0.216	1.755	0.123
-36	60	27	24	63	2.630	0.355	0.213	2.773	0.077
-42	63	27	29	61	2.774	0.355	0.214	2.915	0.074
-48	69	27	28	68	2.227	0.355	0.214	2.368	0.091

Table 28 - Quantitative scores for MTest2 recognition.

[74] The average polyphony of the ideal score varies because it is calculated up to the end of the last note. For lower thresholds some guessed notes overrun the end, making the piece appear longer.

The maximum score is for a threshold of -24 dB. The original file can be heard in (((audio example 2))) and the MIDI resynthesis (for -24 dB), using a piano patch on a Yamaha SY77[Yamaha] is (((audio example 3))). All audio examples are listed in Appendix Q. It will be noted that the current system does not yet assign velocity data.

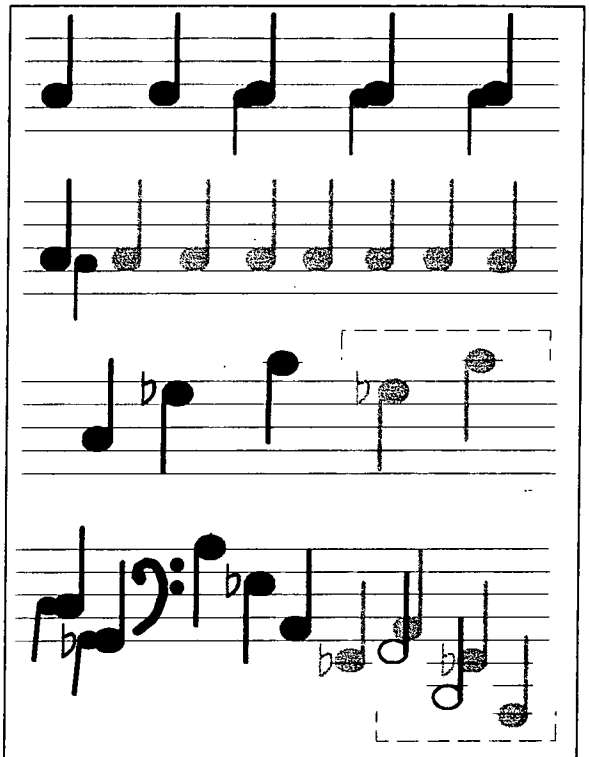


Figure 83 - CPN comparison of original and derived scores for MTest2.

Figure 83 shows the transcription for the threshold of -24 dB in common practice notation. The black notes are the derived notes. Small black notes are short notes added due to the rapid onsets and offsets. Grey notes are those missed by the transcription. The last two low notes are inaccurate in both time and frequency.

The monophonic test pieces have highlighted several potential problems in the analysis. First, the rapid onset transitions cause spurious sines to be extracted. Second, low or short notes are hard to recognise accurately.

7.2.3 Mendelssohn

One piece examined had long been used in the research group as a test piece for the transputer version of Csound[Bailey 90, Bailey 91] – around 30 seconds of the Sonata III for organ written by Felix Mendelssohn between August 1844 and January 1845.[Mendelssohn] The audio was created in mono 16-bit linear format at 32 kHz using Csound files written by Peter Manning. The score and orchestra files are given in Appendix L, and the piece can be heard in (((audio example 4))). Figure 85 shows its logarithmic spectrum. It should thus be noise-free, but has the suspected disadvantage of having no note asynchrony. (Rasch showed that this is an important feature of our ability to perceive simultaneous notes.[Rasch 78]) There are 194 notes on two organ voices, and the maximum polyphony is eight. Note the effect of near-overlapping harmonics – the combination appears to be amplitude-modulated.

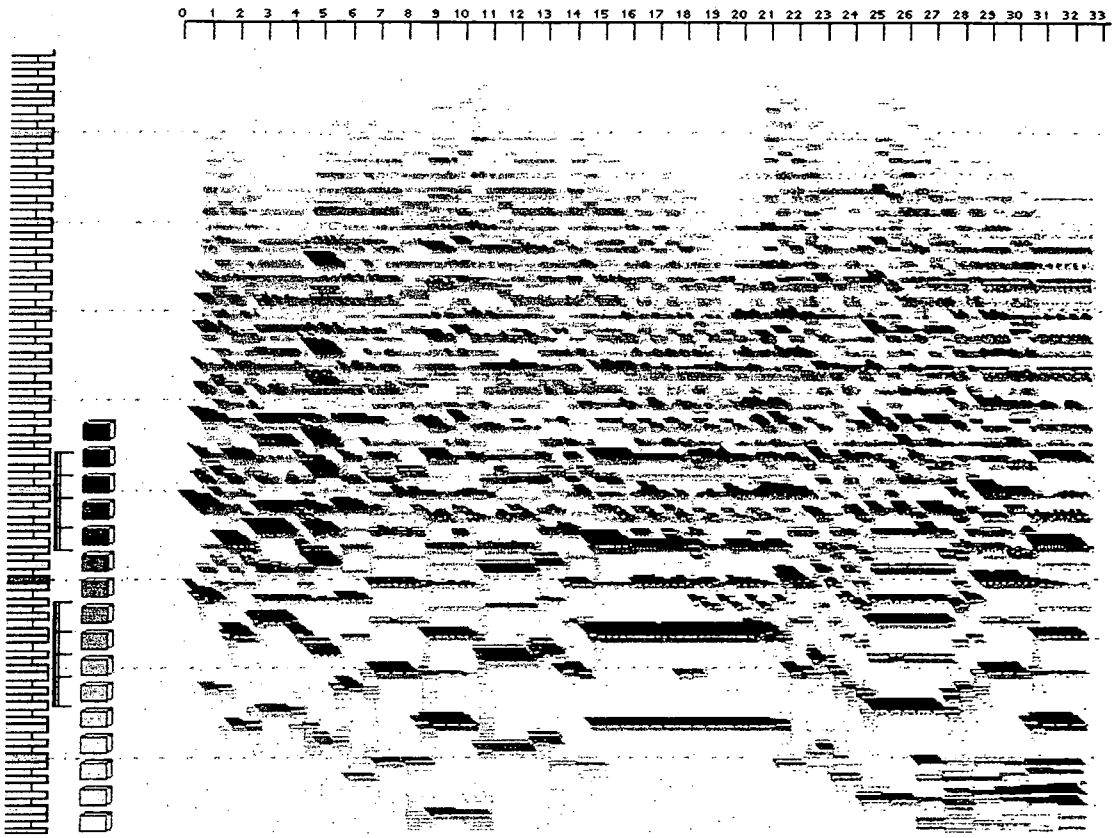


Figure 85 - Logarithmic spectrum of Mendelssohn.

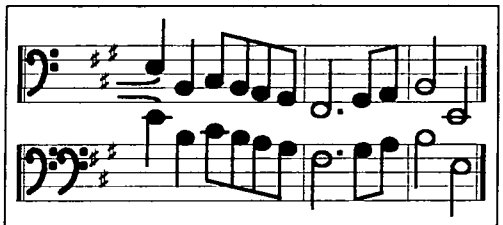


Figure 84 - Doubled bass line of Mendelssohn.

An FFT size of 64 was used. This seems to be suitable, except towards the end, where the quavers in the organ pedal line (doubled an octave below) last for ¼ second at down to 52 Hz, and thus have only 13 cycles. The score of the last three bars is shown in Figure 84, using the bass and sub-bass[Rossing] clefs.

Note also that the spectrum becomes much more complex at the transitions between notes, as it is attempting to model a discontinuity by sinusoids. Due to the scale, it is easiest to see this for the lower notes, but the same effect also occurs at higher frequencies. The analysis uses a Hamming window. The input has an average power of -21.66 dB.

51246 sinusoids were extracted from the spectrum (with flags `-m6 -z48` meaning a threshold of -48 dB and a maximum of 6 sines per block). These were linked into the 6471 partial tracks shown in Figure 86.

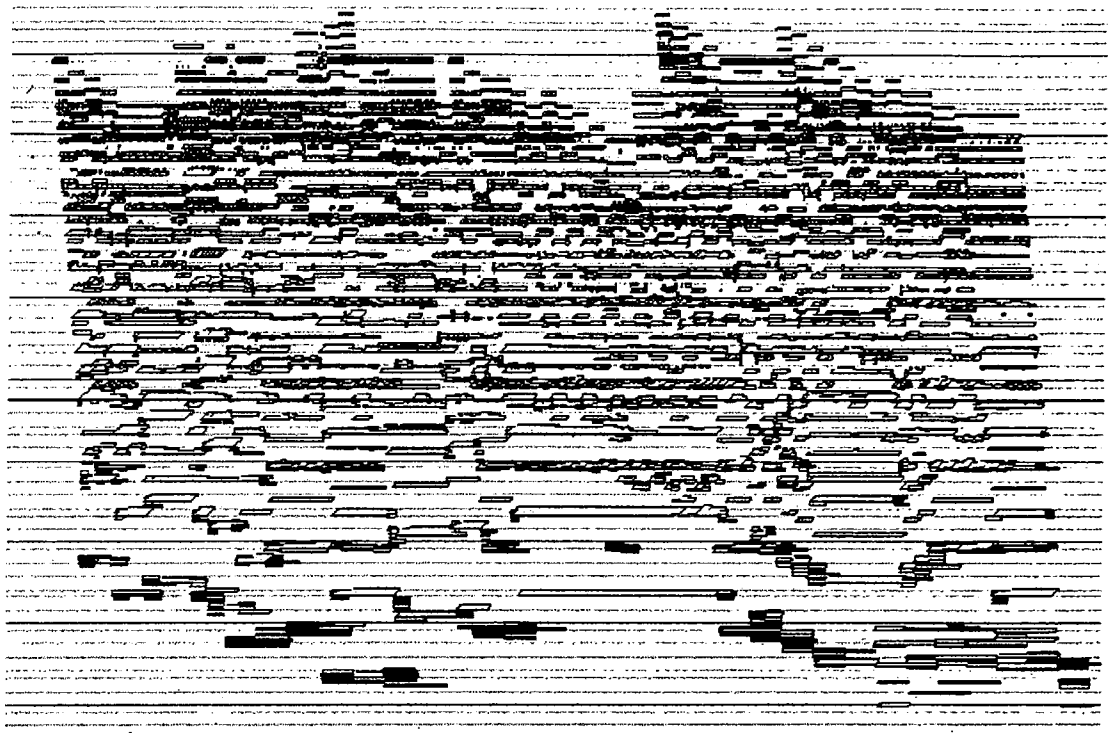


Figure 86 - Partial tracks extracted from the Mendelssohn.

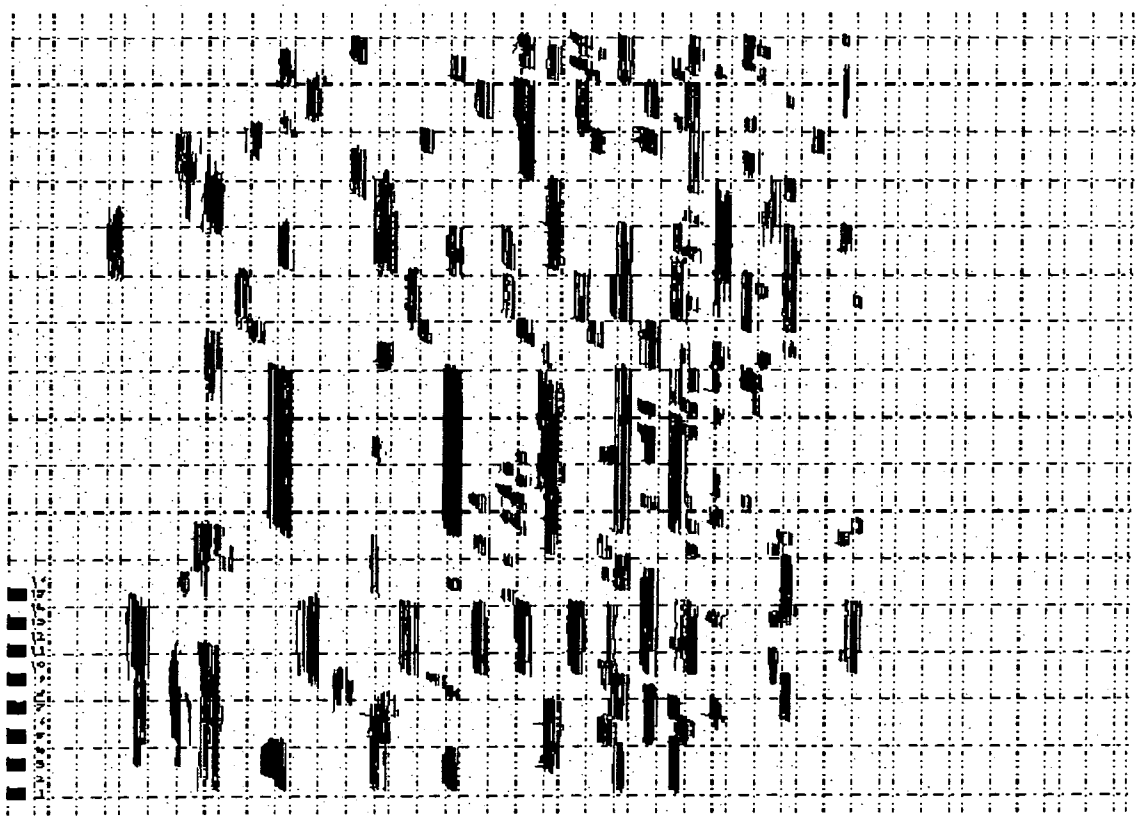


Figure 87 - Battle output for Mendelssohn.

Figure 87 shows the amplitude envelopes of each harmonic of the notes remaining on the battlefield after three iterations. In this figure, time runs from top to bottom and frequency runs from left to right. The scores are compared using READASC, as shown in Figure 88. As before, light grey is expected (or hoped-for) notes, dark grey is predicted notes, and the overlap is in black.

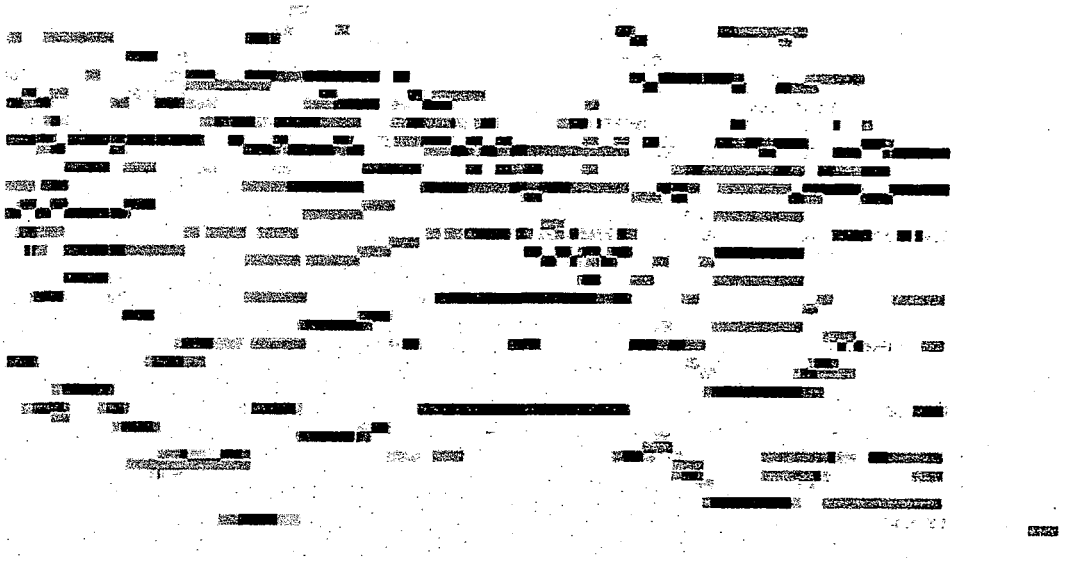


Figure 88 - Score comparison for Mendelssohn.

Next we judge the results using the diagram shown in Figure 89. This shows the overall amount of false positives (extra notes), correct identifications, and false negatives (missed notes).

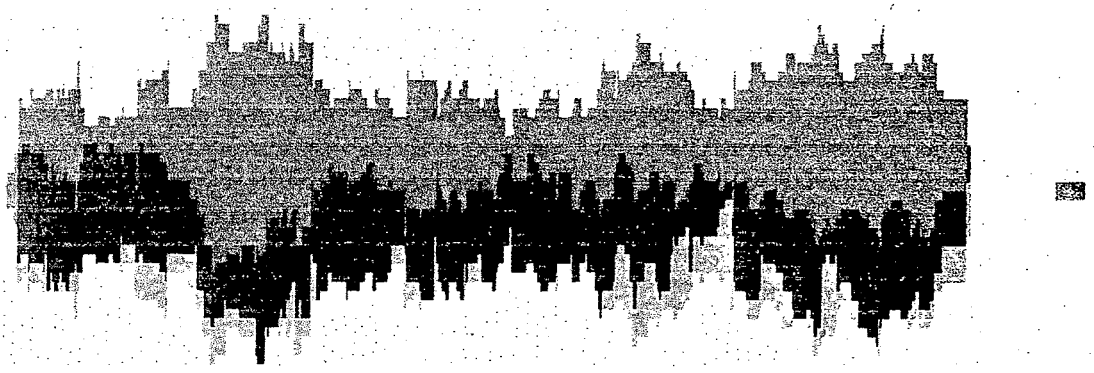


Figure 89 - Evaluation of accuracy for Mendelssohn.

The qualitative scores for the identification are as follows. The polyphony figure is used to give the final score of 39.8%. The 'notes' columns, shown for the test piece, have been dropped as they give misleading information.

	POLYPHONY				
threshold	guessed	ideal	overlap	total	accuracy
-48	9.443	4.998	4.114	10.328	0.398

Table 29 - Quantitative scores for Mendelssohn recognition.

This analysis was repeated for other deconvolution thresholds. Thresholds of -6, -12, and -18 dB gave no output – below are the compared scores for -24, -30, -36, and -42 dB.

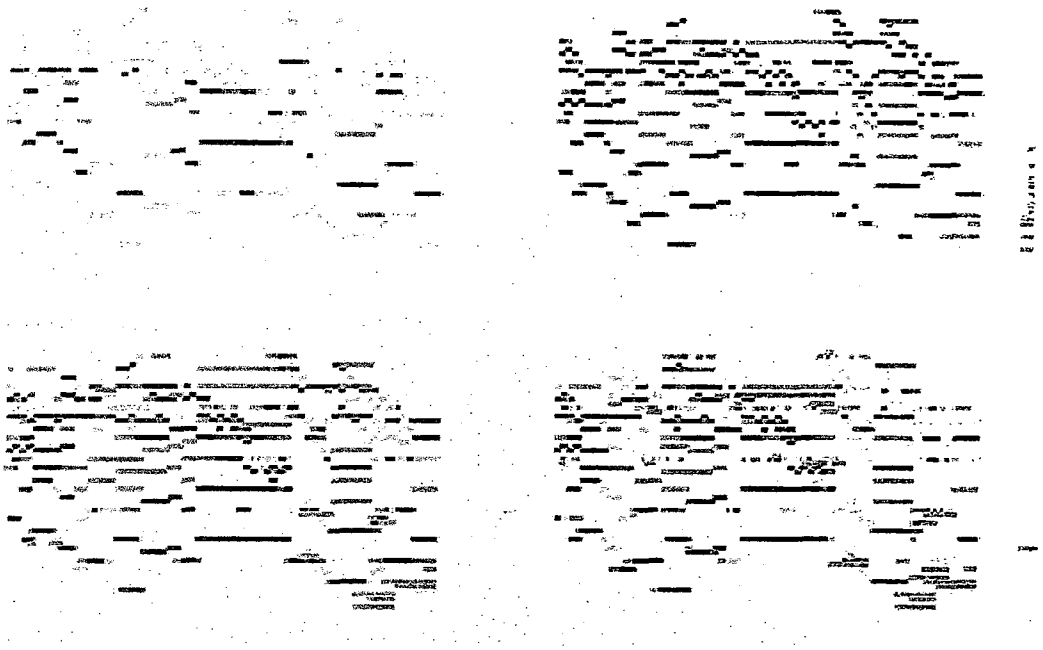


Figure 90 - Comparison of scores for thresholds of -24/30/36/42 dB.

The 'flattened' scores are shown below.

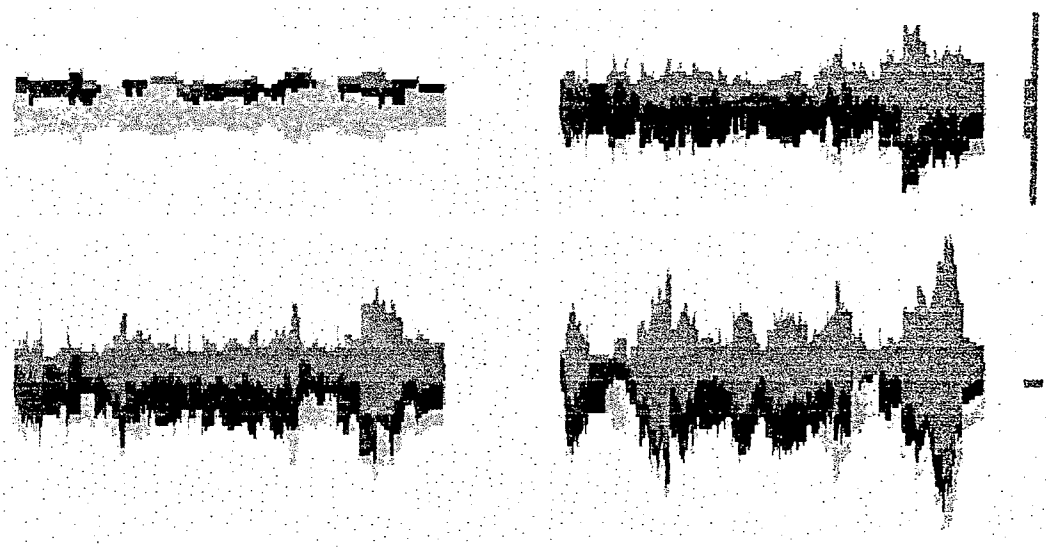


Figure 91 - Comparison of accuracy for thresholds of -24/30/36/42 dB.

The quantitative accuracy is shown in the following table. This also shows the total number of notes, which should be compared to the actual total of 194.

		POLYPHONY				
threshold	notes	guessed	ideal	overlap	total	accuracy
-6	0	0	5.597	0	5.597	0
-12	0	0	5.597	0	5.597	0
-18	0	0	5.597	0	5.597	0
-24	40	1.793	5.597	1.317	6.072	0.217
-30	216	8.078	4.998	4.300	8.776	0.490
-36	169	9.546	5.597	4.264	10.789	0.395
-42	264	10.869	4.931	3.430	12.370	0.277
-48	*220*	*9.443*	*4.998*	*4.114*	*10.328*	*0.398*

Table 30 - Quantitative scores for Mendelssohn recognition.

The -48-dB line used different analysis parameters to the others, as it was otherwise too big to be processed. The original deconvolution by PICKOUT used the flag `-m6` rather than `-m20`. This sets the maximum number of sinusoids to be removed from each spectrum. This has the effect of reducing the number of sines removed when noise or a discontinuity is present in that block. However, the fact that it appears to improve the accuracy may suggest that it is a powerful parameter in the analysis.

Another way to judge the effectiveness of the transcription is of course to listen to the results. The original is (((audio example 4))), and (((audio example 5))) is the score transcribed using a threshold of –30 dB and played on the SY77. This figure gave the highest 'accuracy', and is also the most audibly similar to the original.

It is also possible to compare the scores visually, although the erratic nature of the transcription into common practice notation may prevent a clear comparison for such a complex example. The scores are compared in the figure below, for the threshold of -30 dB. The CPN 'rendering' is done by Cakewalk Professional for Windows, and has been quantised to quavers and put into the correct key.

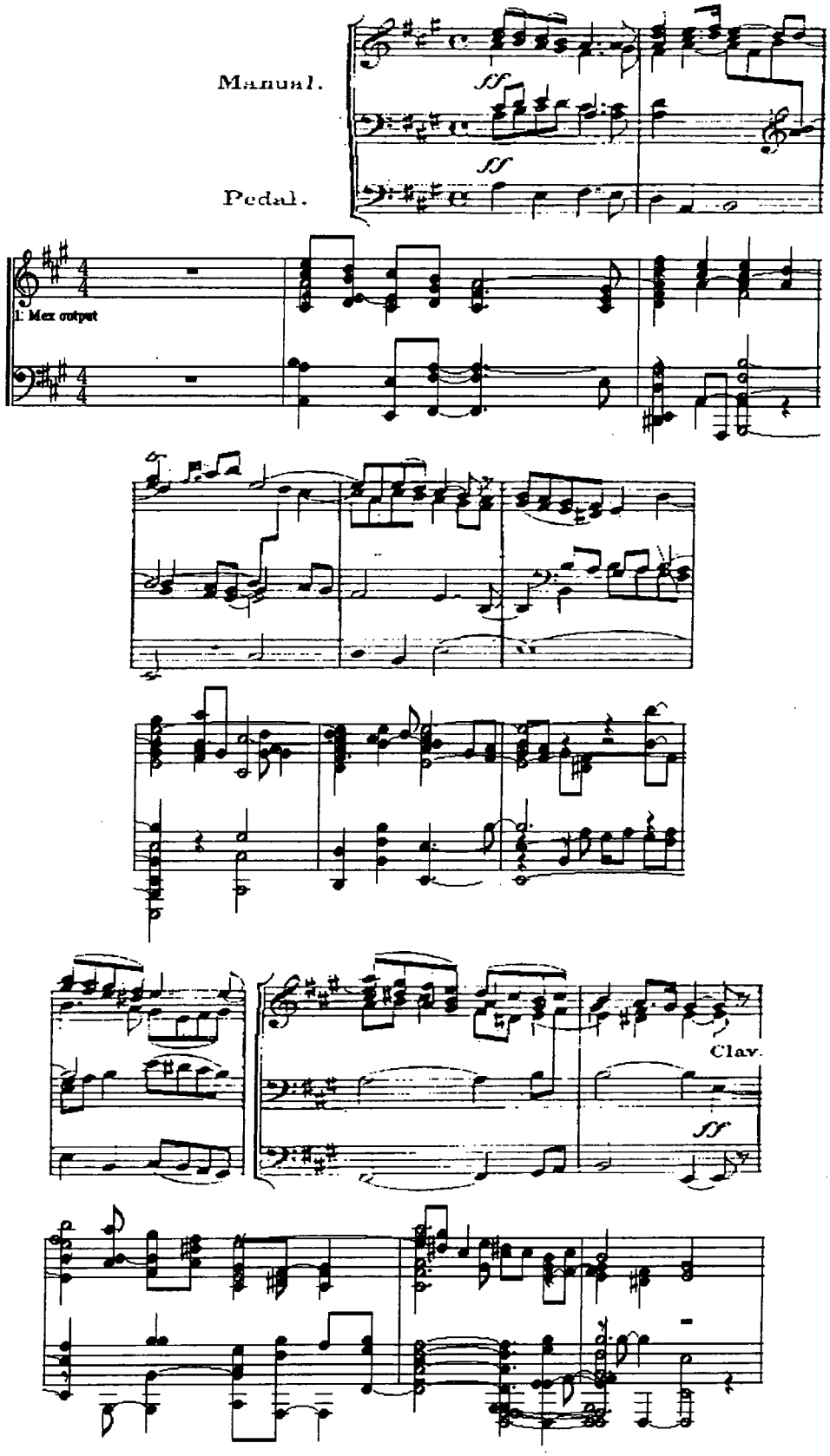


Figure 92 - Comparison of CPN scores for the Mendelssohn.

It is worth comparing the comparison methods. It is important to note that in all cases, timbres are neither evaluated or conveyed.

Comparison Method	*Advantages*	*Disadvantages*
Note count	Simple	Simplistic
Piano-roll comparison	Intuitive	Needs score
Quantitative overlap	Gives single figure	Needs score
Common Practice Notation	Familiar	Needs timing information
MIDI Resynthesis	Intuitive	

Table 31 - Comparison of comparison methods.

There are many different types of error.

Notes added from harmonics

One common error is to have extra notes corresponding to harmonics of a correct note. This is because each track is viewed as potentially a note, and notes may share partials.

Notes connected

Another problem is for legato notes to become connected. Even if there is no actual overlap, notes that are separated by less than two block sizes may be seen as connected. This would be exacerbated by reverberation.

Tracking at octaves

There are also problems in tracking sinusoids near the boundaries between octaves, which give rise to spurious or missing B's.

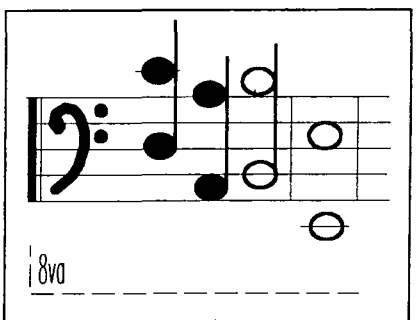


Figure 93 - Bars 2-3 of the Mendelssohn.

Harmonics wrongly connected

In the case shown in Figure 93, the third, sixth, ninth, ... harmonics of the low E were recognised as a continuation of the upper B, rather than belonging to the low E.

Splashed notes

There is often a 'splash' of notes up to a few semitones from the correct note, and this is most noticeable for very low notes. A rapid onset causes a spread of spectral energy not only around the fundamental, but also around other harmonics. These can be combined into seemingly viable notes. Sudden changes in the spectrum cause problems over a block; at higher frequencies, these are eliminated by the minimum note length, but at very low frequencies these are extended to the block size, which is larger.

Bass too early

The bass line audibly 'anticipates' the times. This is because when a sinusoid starts part-way through a block, the time output is the start of the block. This on average shifts the times back by half the block size, and this is significant at the low sample rates of the lower octaves.

7.2.3.1 Summary

The principal advantage of the multirate approach is the fact that much of the bass line has been recognised accurately, even though the semitones are only a few Hertz apart. The MIDI rendition captures many of the details are recognisable, even for the relatively high polyphony.

7.2.4 Poulenc Sonata for Horn, Trumpet, and Trombone

This piece was composed in 1922 by Francis Poulenc (1899-1963).[Poulenc] The last section of the first movement was used – it lasts around a minute, and the start forms (((audio example 6))). The original source was a record by the Philip Jones Brass Ensemble. The record had been copied to cassette tape about ten years ago. A further cassette copy was played on a personal cassette recorder and sampled at 16000 Hz by the Gravis UltraSound card. Due to the multiple generations, and to poor equipment fidelity, the resultant input had a high level of noise. The average signal level was -15.28 dB.

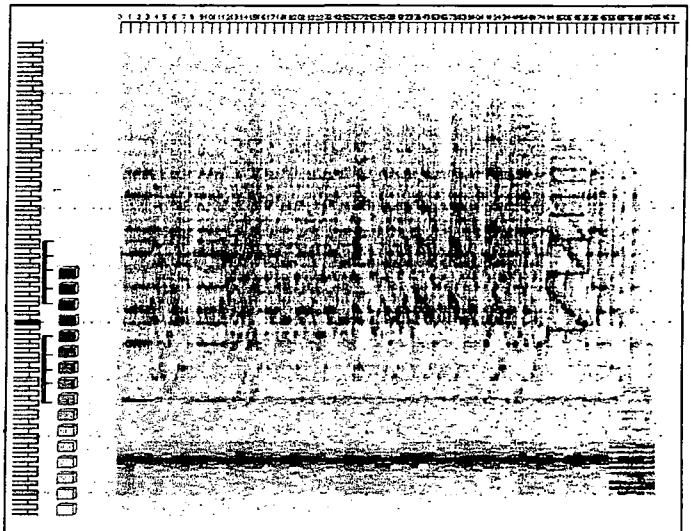


Figure 94 - Spectrum of Poulenc.

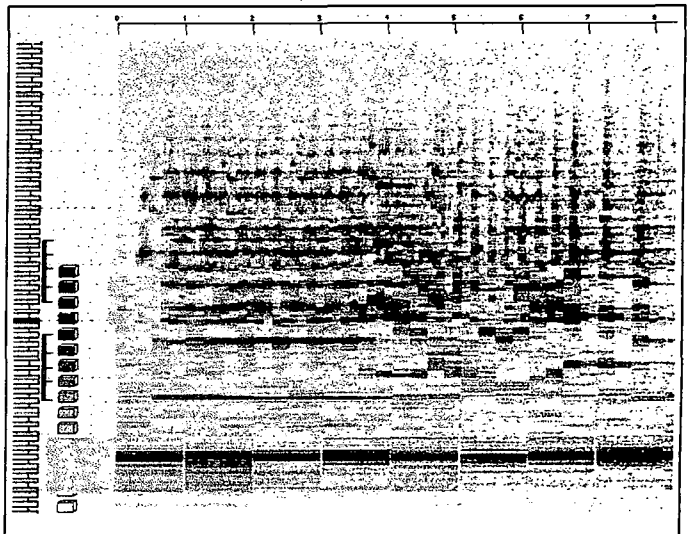


Figure 95 - Spectrum of start of Poulenc.

The spectrum of the piece is shown in Figure 94. Since the total time is around a minute, it is impossible to derive much detailed information from this picture. One obvious feature is the presence of some rather severe mains hum. In fact, the mains hum appears to have some 2nd-harmonic power too.

Figure 95 shows an excerpt from the start in more detail. This eight seconds contains 33 trumpet notes, 31 horn notes, and 18 trombone notes, a total of 82.

The issue of the number of periods in a note has been discussed before. The Poulenc piece is played at a lively tempo, and has many short notes. It would be reasonable to suggest that many notes are no longer than 0.1 seconds. If a 100-ms note is at a frequency of 200 Hz, then there will only be 20 cycles of the wave. An approximate count of the number of notes in the Poulenc is:- trumpet 220, horn 173, trombone 118, for a total of 511. The score (in C) of the short extract above is shown in Figure 96.

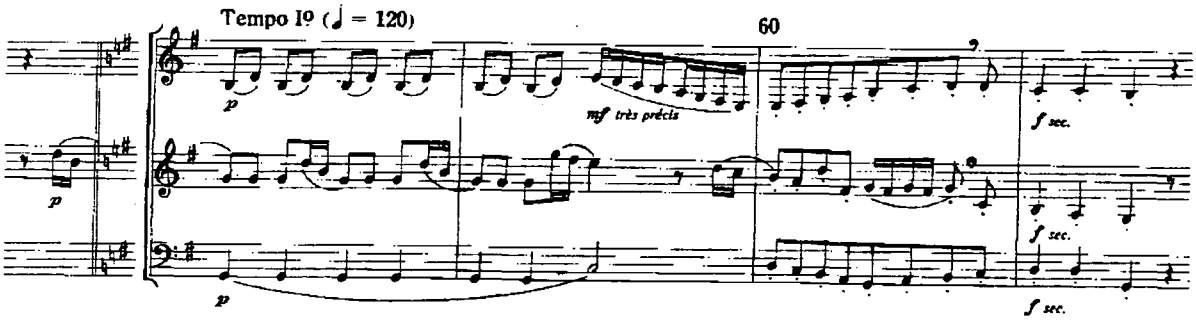


Figure 96 - Score of start of Poulenc.

Deconvolution was tried at various thresholds, but none of them was able to even remotely identify the notes. Audition of the resultant transcription could also not be identified with the original. As the precise score was not available, it was not possible to quantify the accuracy.

Several problems are highlighted by this piece. The very short notes have been discussed earlier. The strong mains hum claims harmonics belonging to the real instruments and appears to be a stubborn pedal point (in fact it is transcribed as several overlapping notes at neighbouring frequencies). In addition, no trombone notes were identified. This may be because fundamentals of low trombone (and horn) notes can be very weak.

Even if these problems could be solved, one final question relates to the source separation process. Our analysis system has tried to break the input into individual notes. How could we then categorise these notes as belonging to one of the three instruments, particularly in this case when the timbres are fairly similar?

7.2.5 Schumann Träumerei

Träumerei ('reverie' or 'dreaming') was written in 1838 by Robert Schumann (1810-1856). A score of the extract is shown in Figure 97, and the example can be heard in (((audio example 7))). The written polyphony is six, although for the piano, the effects of the sustain pedal and legato phrasing can mean that the polyphony is higher than written. There is a total of 52 notes.

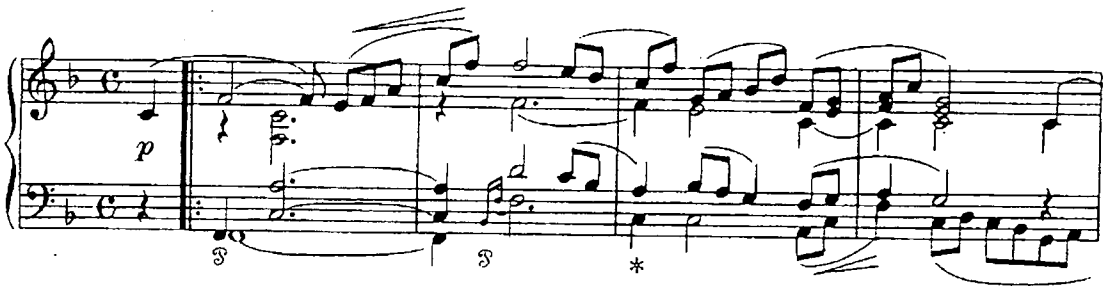


Figure 97 - Score of Träumerei.

It was sampled from CD and converted to mono with a sample rate of 32000 Hz. The test for this piece was that the pianist was known to be one of the 28 mentioned in a previous study on expressive timing.[Repp] The aim was to determine which of the pianists it was. The spectrum is shown in Figure 98. As the aim was deduction of timing information, a smaller FFT, of size 32, was used.

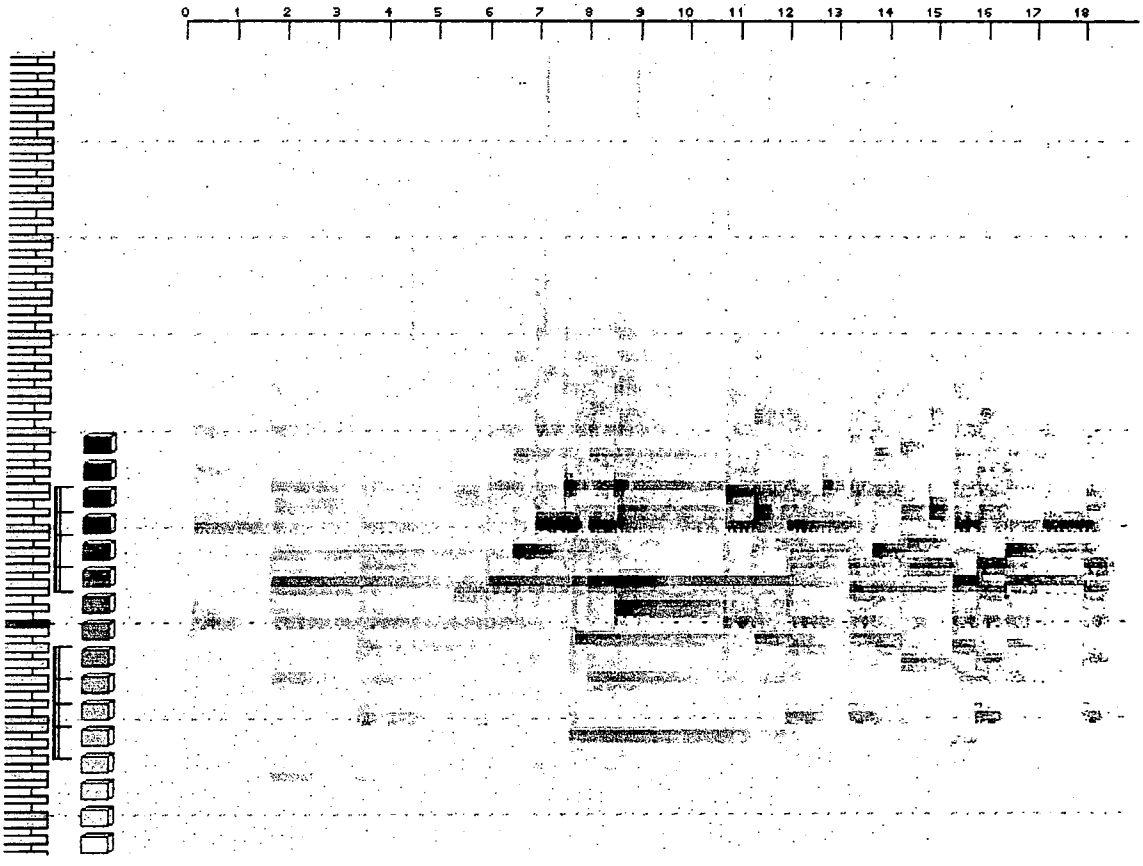


Figure 98 - Spectrum of Träumerei.

As with the MTest examples, eight different thresholds were used. The average level was -19.64 dB. Pickout (v0.24) used the flags `-m20` (up to 20 sines removed from each FFT) `-n32` (FFT size) `-a` (automatic mode – don't ask the user to press keys). The battle (v0.27) used the flags `-t0.2` (minimum note length 0.2 s) `-nk` (no keypresses, same as the auto mode in pickout).

threshold	*sines*	*partials*	*trivial*	*short*	*bad harm.*	*quiet*	*left*
-6	32	13	7	6	0	0	0
-12	109	37	20	15	2	0	0
-18	352	113	55	43	15	0	0
-24	1039	339	110	173	52	0	4
-30	2834	947	205	597	138	0	7
-36	6170	2039	330	1428	250	1	30
-42	11892	4030	507	2955	431	24	113
-48	20140	6781	740	5012	655	80	294

Table 32 - Summary of analysis of Träumerei.

The above table summarises the Träumerei analysis.

In order to quantitatively judge the analysis, we need the 'correct' score, i.e., the correct timings, which is not available. This was derived approximately, using an interactive method described below. The first three threshold values gave no output.

Figure 99 shows the comparisons of the files, for thresholds -18, -24, -30, -36, -42, and -48 dB. One problem arises with the display: the colours are exclusive-or'ed on screen. This means that two identical notes will rub each other out.

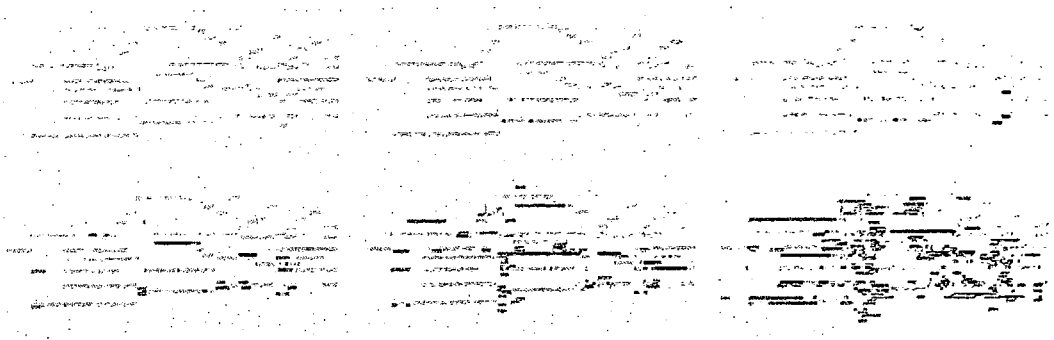


Figure 99 - Comparison of scores for Träumerei.

Figure 100 is the 'flattened' diagram showing the polyphony matched. The overlap can be seen to be minimal.

Figure 100 - Comparison of polyphony for Träumerei.

Below is a table of the accuracy.

		POLYPHONY				
threshold	notes	guessed	ideal	overlap	total	accuracy
-6	0	0	3.918	0	3.918	0
-12	0	0	3.918	0	3.918	0
-18	0	0	3.918	0	3.918	0
-24	4	0.232	3.918	0.134	4.016	0.033
-30	7	0.411	3.918	0.236	4.093	0.058
-36	30	2.691	3.918	1.059	5.550	0.191
-42	113	6.926	3.918	1.586	9.261	0.171
-48	294	16.05	3.918	2.171	17.796	0.122

Table 33 - Quantitative scores for Träumerei recognition.

The scores are much poorer for this example than for the synthetic organ. The resultant MIDI rendering can be heard in (((audio example 8))). The transcription was not good enough for the identity of the pianist to be decided, so an approach similar to that used by Repp was taken. It is unfortunate to note that the automatic analysis method failed the task whereas the interactive method worked with sufficient accuracy.

7.2.5.1 Interactive analysis

Using Windows sound file editors, I calculated the times both by auditioning short samples and by examining the spectrogram. The main difficulty with this method is that it is very difficult to distinguish near-simultaneous onsets, as noted by Repp. In addition, it is much more difficult to determine the durations. However, this was enough to be able to distinguish the times sufficiently accurately to determine the pianist. The full analysis is given in Appendix K.

The times and pitches were made into a Cakewalk ASC file and then a Cakewalk work file. This was then edited to give the notes a 'reasonable' duration to allow the score comparison above.

7.2.6 Grieg Piano Concerto

The opening of this well-known concerto[Grieg 68] was copied from CD to tape, then sampled at 16000 Hz by the GUS card. The score is shown in Figure 101 and the spectrum is shown in Figure 102. The example forms (((audio example 9))).

There are 166 notes in this excerpt, not including the initial timpani roll and orchestra hit. The written piano polyphony is often 8, and has a maximum of 17 during the arpeggio in bar 4 played with the sustain pedal down.

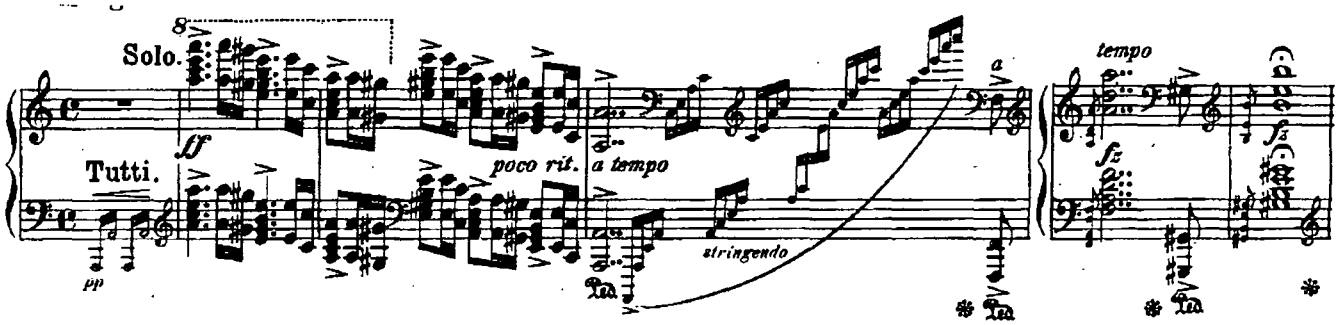


Figure 101 - Score of start of Grieg Piano Concerto.

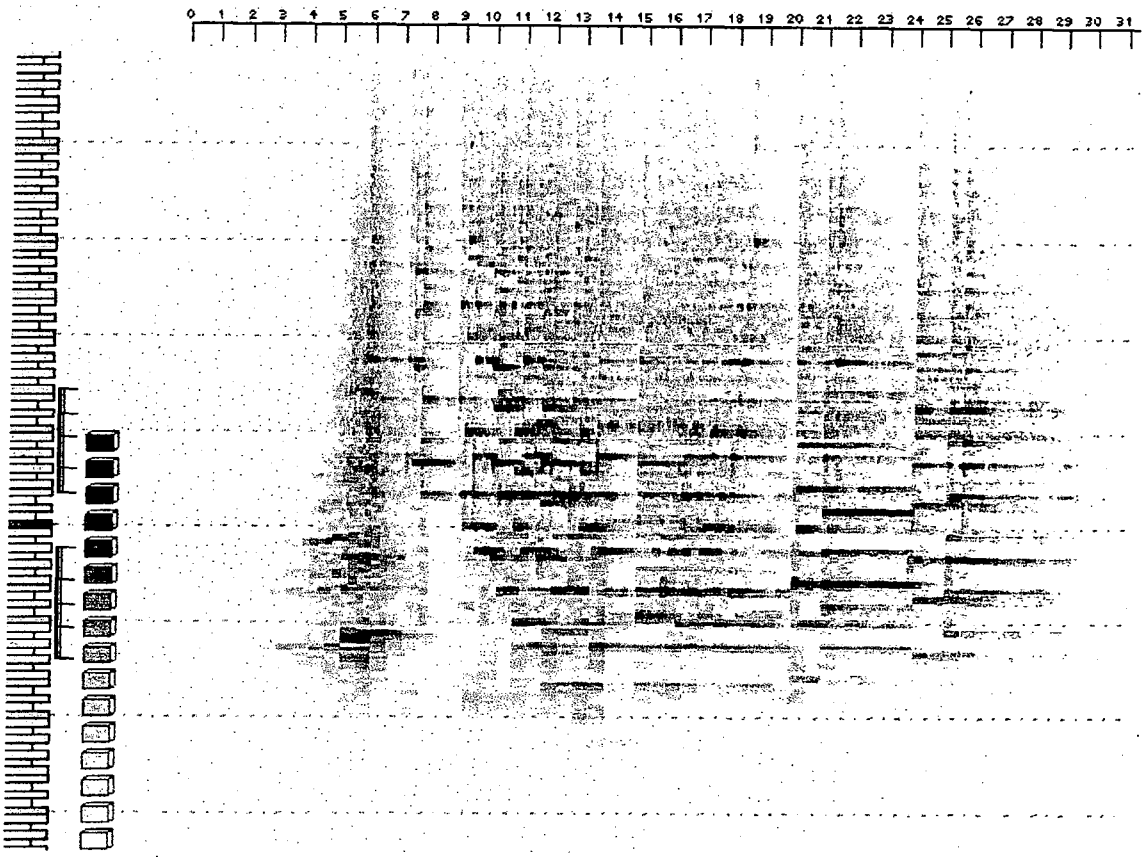


Figure 102 - Spectrum of start of Grieg Piano Concerto.

The initial timpani roll raises two questions. First, although the timpani has inharmonic modes, some of them do form a very roughly harmonic sequence – is this enough to derive a pitch? Second, given the effect of natural reverberation and the poorer time resolution, will the notes be distinct? The orchestra hit raises one more difficulty – we are unable to distinguish each note, so what is the correct transcription?

The extract has an average power of -18.70 dB. The sines were extracted using the `-m6` flag, which limits the number picked per octave to 6. The battle used the flag `-t0.05` to set the minimum length of a note to 50 milliseconds. The results for several thresholds are shown below.

threshold	*sines*	*partials*	*trivial*	*short*	*bad harm.*	*quiet*	*left*
-6	46	31	28	0	3	0	0
-12	440	202	†				
-18	1911	744	354	120	225	3	42
-24	5443	1772	733	398	533	23	85
-30	10476	3155	†				
-36	16901	5003	1814	1828	1037	125	199

† – Bugs unresolved at the time of writing prevented the battle running for these thresholds.

Table 34 - Summary of analysis of Grieg Piano Concerto.

On auditioning the output, there were few points where the transcription was accurate. The MIDI resynthesis for -18 dB can be heard as (((audio example 10))). The CPN for this output is shown in Figure 103. As the timing information is not available, no attempt has been made to set suitable barlines. Some of the descending line, and some later chords, have been captured, but most of the details have not been recognised.

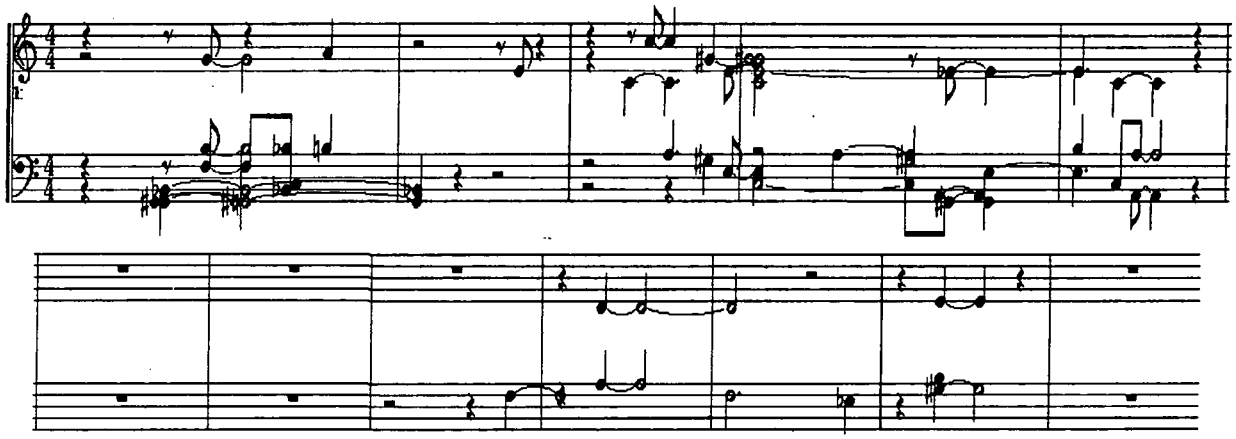


Figure 103 - Transcribed score of Grieg Piano Concerto.

7.2.7 Death of Aase

The first four bars of this piece, from "Peer Gynt", were copied from the same CD as the Grieg Piano Concerto described above, and sampled at 16000 Hz.[Grieg 76] It forms (((audio example 11))). Since the signal level is so low, the effective SNR of the source is poorer. The score is shown in Figure 104, and

the spectrum, which uses an FFT of size 64, is shown in Figure 105. It has an average level of -16.61 dB. There are 64 notes in total, counting sections as notes.

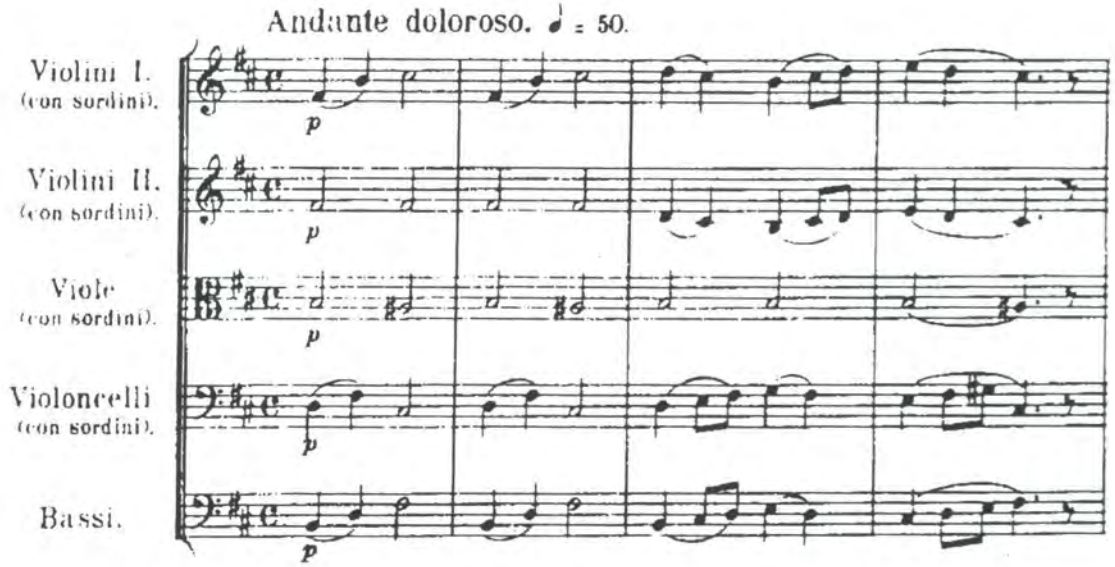


Figure 104 - Score of 'Death of Aase'.

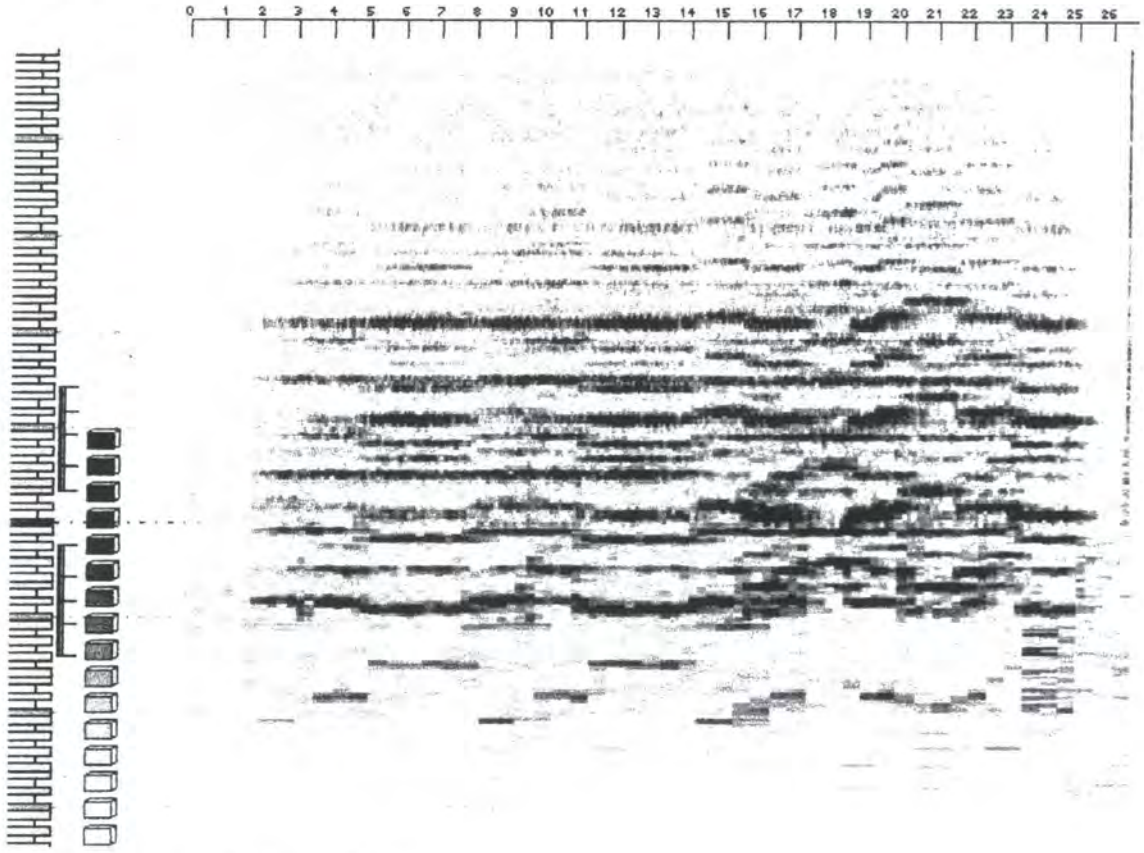


Figure 105 - Spectrum of 'Death of Aase'.

This was analysed using eight thresholds in Pickout (v0.23), with a maximum of 20 sines per FFT. As we know that this has relatively long notes, we give the analysis a helping hand by specifying -t0.7 to eliminate notes shorter than 700 ms.

The results are shown in the table below. Experiments with lower thresholds could not be completed due to memory problems.

For each part there are up to 16 players. We can assume that their vibratos and other parameters are independent of each other. This results in a complex envelope for each partial.[75] Although we know from experience that there are sixteen first violins, we cannot separate them. This example illustrates a difficulty in the definition of the transcription task – is the correct transcription that which models our perception of 'the sound of a violin section', or is it that which separates this into 'the sounds of sixteen individual violins'? If the former, we can ask how violins and other instruments fuse together; if the latter, we can ask how a computer model could be constructed so as to far exceed the capabilities of our processing.

threshold	*sines*	*partials*	*trivial*	*short*	*bad harm.*	*quiet*	*left*
-6	125	86	53	33	0	0	0
-12	741	331	197	130	2	0	2
-18	2277	734	355	349	23	0	7
-24	6992	2426	895	1449	31	2	49
-30	27014	11628	5778	5644	9	46	151
-36	70072	22622					
-42	123067	-	-	-	-	-	-
-48	137961	-	-	-	-	-	-

Table 35 - Summary of analysis of Aase.

The best appears to be for -24 dB. This is shown in CPN in Figure 106 and can be heard in (((audio example 12))).

Again, the transcription is poor. As the 'correct' score is not known, it is not possible to quantify the accuracy.

[75] See also Dubnov's work comparing the bicoherence of viola sections and violas.[Dubnov]

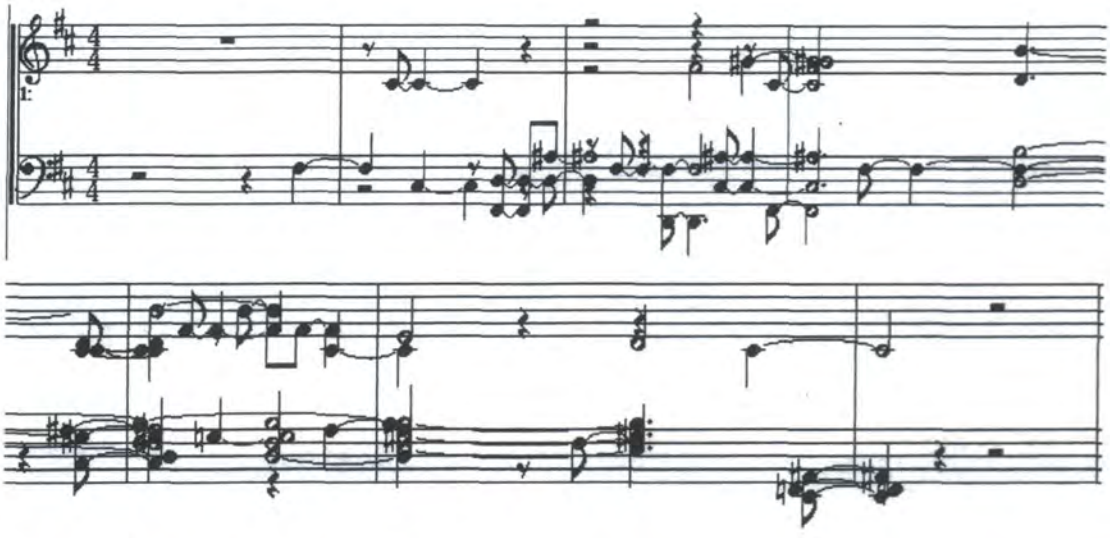


Figure 106 - Transcribed score of Death of Aase.

7.2.8 Didgeridoo

The original performer is a former flatmate, Martin Perlbach. His performance, consisting of a loud deep breath in followed by a 12-second note, was recorded on cassette, and a copy of this was sampled by the GUS at 16000 Hz. The spectrum is shown in Figure 107, and the input is (((audio example 13))).

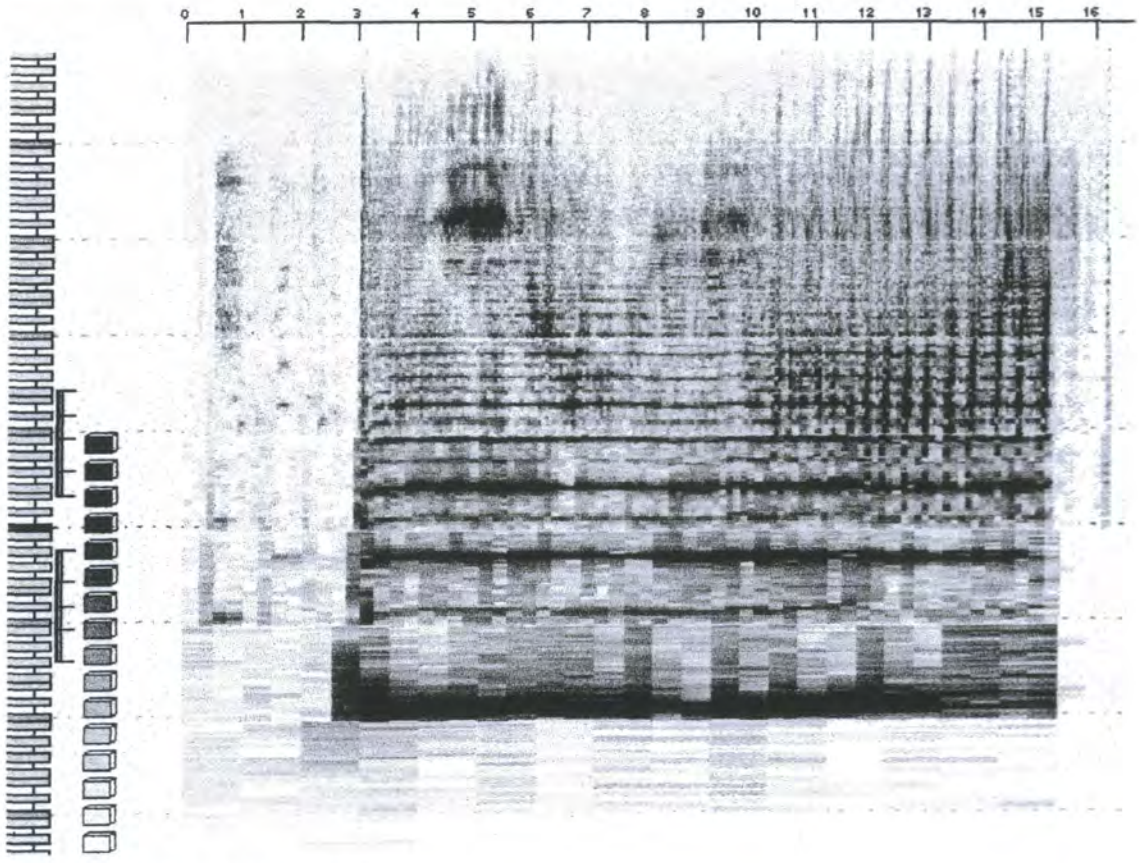


Figure 107 - Spectrum of Didgeridoo.

The average level is -15.51 dB. The analysis was done using Pickout (v0.24) with a threshold of -24 dB and a maximum of 6 sines per FFT, TrakSin (v0.08), and FT (v1.05). The battle (v0.29) used a minimum note length of 0.5 seconds.

threshold	*sines*	*partials*	*trivial*	*short*	*bad harm.*	*quiet*	*left*
-6	235	151	78	70	3	0	0
-12	1487	595	179	378	27	0	11
-18	4676	1893	484	1335	30	4	40
-24	9948	3964	935	2950	23	4	52
-30	13964	5779	1323	4362	31	4	59

Table 36 - Summary of analysis of Didgeridoo.

The score comparisons in Figure 108 are for thresholds of -12, -18, -24, and -30 dB, and Figure 109 shows the flattened comparison.

Figure 108 - Score comparison for Didgeridoo.

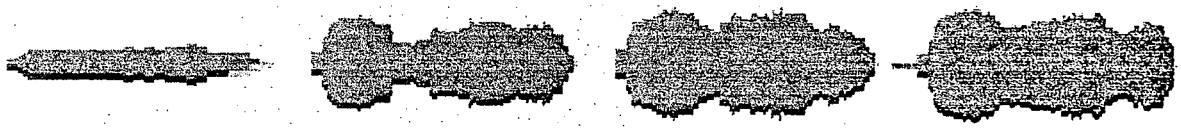


Figure 109 - Flattened score comparison for Didgeridoo.

The ideal transcription is a single note, on C#2 (69 Hz). There are long notes similar to this, although there are actually several overlapping notes, including some at C2 and D2. Many of the added notes are at G#3, E#4, etc., corresponding to the third, fifth, etc. harmonics of the fundamental. As can be seen from the spectrum, the odd harmonics are considerably stronger than the even harmonics. This is to be expected because while the didgeridoo is a lip-reed instrument, it has a cylindrical bore and should thus have comparable modes to a clarinet.

It is very simple to form the ideal score. The quantitative evaluation is as follows:-

		POLYPHONY				
threshold	notes	guessed	ideal	overlap	total	accuracy
-6	0	0	0.836	0	0.836	0
-12	11	3.726	0.836	0.696	3.866	0.180
-18	40	9.391	0.836	0.828	9.398	0.088
-24	52	11.432	0.836	0.828	11.440	0.072
-30	59	12.043	0.836	0.828	12.051	0.069

Table 37 - Quantitative scores for Träumerei recognition.

This example highlights the large amount of control information that can be present in a note. The 3-Hz fluctuations in all of the harmonics (corresponding to the simple physical control of tongue-palette distance) cannot be encoded efficiently. The desire to summarise information about several harmonics is part of the reasoning behind group additive synthesis, which uses a reduced number of envelopes.

7.2.9 Ringdown

Another piece examined was the ringdown of Durham Cathedral bells. The ringdown refers to the process of returning the bells from their inverted normal striking position to hanging vertically. This is not considered part of the 'performance', but still requires skill, especially for the players of the larger bells. The score would consist of a ten-note descending major scale played repeatedly, but with decreasing inter-note and inter-scale times. This is shown *very* schematically in Figure 110. The intention in transcription was stimulated by discussion with Ian Breakwell, the artist-in-residence at Durham Cathedral, who wished to develop an animation or graphic based on the sound of the ringdown.

Figure 110 - Pseudo-score of ringdown.

This was originally recorded on DAT at 48000 Hz by Ron Geesin and resampled by the CardD. The spectrum of a short extract is shown in Figure 111. The sound forms (((audio example 14))).

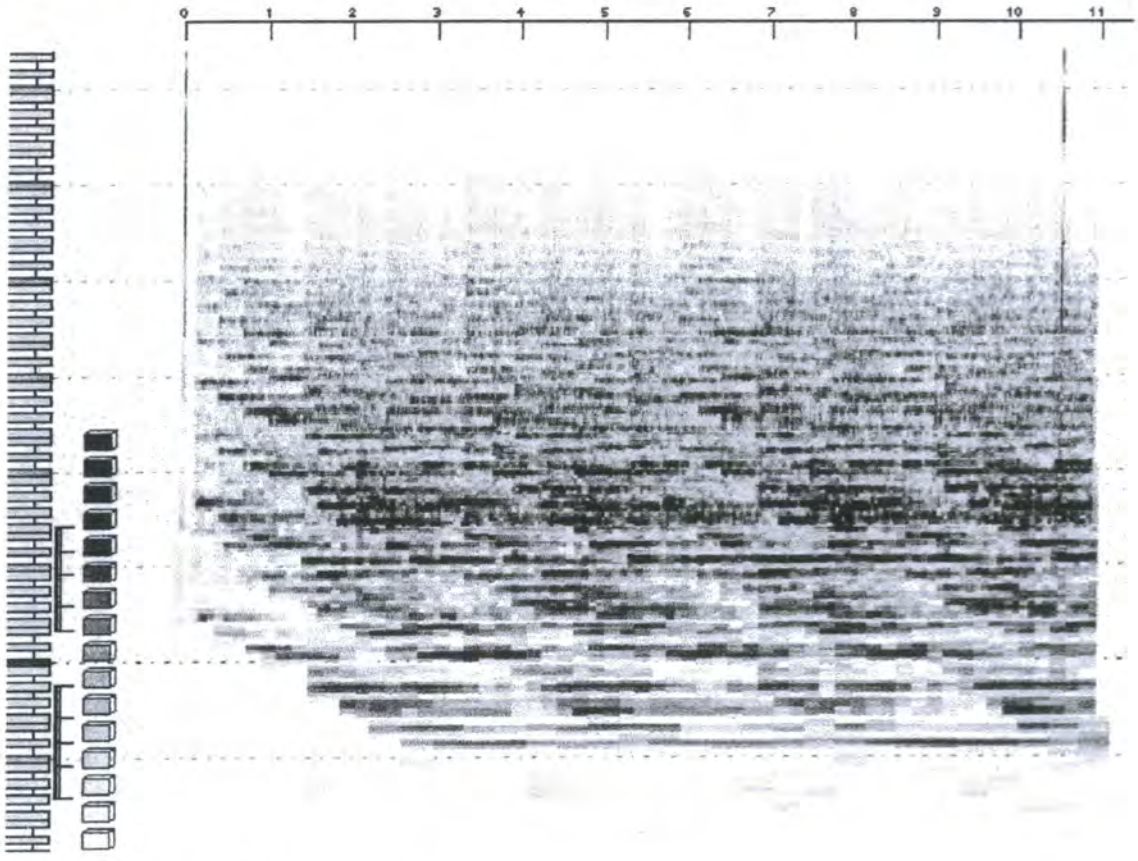


Figure 111 - Spectrum of start of ringdown.

One fact prevented a complete analysis – the size of the input file was around 8 minutes in all. Although virtual memory was used in several subsystems, the requirements for data that must remain in real memory were still too high. Even when data is in virtual memory, the pointers to that data would typically be in real memory. Of course these pointers could be put into virtual memory but only with a great degradation of speed.

The harmonic matcher only looks for harmonics at near-harmonic multiples of the fundamental frequency. However, the bell has inharmonic modes of vibration that have combinations of radial and longitudinal nodes.[76] The *approximate* relative frequencies of its vibration modes are 0.5, 1, 1.2, 1.5, 2, 2.5, 3, and 4. Rossing uses the names hum, prime, minor third, fifth, octave, upper third, twelfth, and upper octave; Sundberg uses the names hum, strike, tierce, quint, nominal, deciem, duodeciem, and double octave. **[Rossing, Sundberg 91]**

However, some of the modes are nearly harmonic, so it is informative to see if these are sufficient, by analysis of the above short segment. The results are shown below.

threshold	*sines*	*partials*	*trivial*	*short*	*bad harm.*	*quiet*	*left*
-6	37	25	25	0	0	0	0
-12	547	250	250	195	46	9	0
-18	2730	1064	†				
-24	7136	2357	1043	1130	121	0	63
-30	11493	3464	1143	2093	122	6	100

† The reordering stage failed for this threshold.

Table 38 - Summary of analysis of ringdown.

The output score for a threshold of -24 dB is shown in Figure 112. It shows that the descending scale has been partly recognised but that many notes have been added corresponding to other partials.

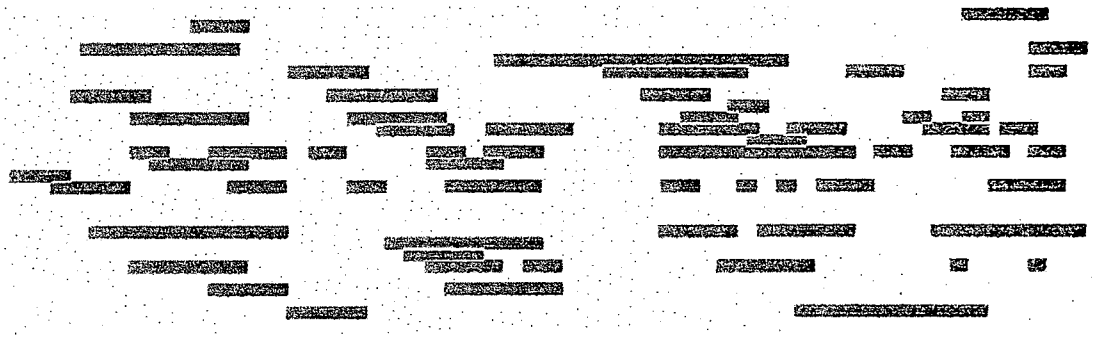


Figure 112 - Score output for ringdown.

[76] A bell is topologically similar to a circular membrane, and the modes of vibration form similar 'dartboard' patterns. However, the density profile is much more complex and the bell is clamped at the centre rather than the edge.

7.3 Limitations of the current model

Many transcription errors have been highlighted above. MTest suffered from rapid onsets/offsets, the Mendelssohn has high polyphony and no onset asynchrony, the Poulenc has missing fundamentals, short notes, and noise, the Schumann is quiet, the piano concerto is too polyphonic, Aase is quiet and has combined strings. The didgeridoo has complex control information, and the ringdown is inharmonic. The performance is arguably the best for the input it was originally designed using, the Mendelssohn.

Each stage of the process is implemented as a separate program, and data is passed via temporary disk files. This is a practical measure on the PC architecture, due to the large amount of memory required during each stage for the input and output data and the program itself.

However, it also means that the various parts of the model cannot interact. Initially, all the data is in the form of a waveform. The C40 analysis converts this to spectra without knowledge of the onset times. The deconvolution converts spectra to sinusoids without knowledge of the sinusoids in consecutive blocks. The frequency tracker links sinusoids without knowledge of what note they might belong to. The battle process simplistically removes everything that *doesn't* look like a note. The data flow is wholly *bottom-up*. Criticisms of this approach have been raised in previous research reviewed earlier.[BrownG 94a, Slaney 95b, Cooke 96a]

A more effective approach may demand an architecture where all the processes can interact, so that the subsystem that determines that a partial is present can ensure that the deconvolver looks specifically for that frequency. However, a single PC is not currently sophisticated enough to supply limitless memory and seamless multitasking.

Features that are smaller than the block size cannot be recognised easily, and features much longer cannot be coded more efficiently. The frequency/time resolution is poor for short low notes – this was found in the Poulenc, the Mendelssohn, and the test pieces MTest1/2. The choice of a fixed Q also means that in periods of relative spectral inactivity, we cannot capitalise on this.

As a result, the hoped-for compression did not materialise – the derived additive-synthesis envelopes are more cumbersome than the original data, and at several stages data is discarded to alleviate the high memory requirements. As well as being bulky, the eventual representation becomes more complex and less intuitive. What we turn audio into is a set of notes consisting of a set of braids consisting of a set of partials consisting of a sequence of multirate additive synthesis control structures. It would takes a large amount of computation to turn it back into a wave, and the advantages of the multirate FFT would no longer be available since we have interpolated frequencies other than those that can be computed by the reverse process. Resynthesis was never implemented.

Throughout the processing, there are many thresholds that can only be set by trial and error. Many of the thresholds were set to whatever appeared to work on the Mendelssohn.

The transcription process clearly has problems separating the sources. It is also not possible to compare the apparent timbres of several notes.

The system is ill-suited to inharmonic timbres, noise, and missing fundamentals. In the case of the cathedral bells, it is the synchronisation of simultaneous onsets that fuses the partials into a single tone, yet this cannot be examined, as the current model only groups partials by harmonicity. The problem of the missing fundamental has also not been solved. We cannot perform analysis on stereo files. While we could of course process two files separately, this does not allow us to easily combine information from the two channels.

The importance of graphical output has led to most stages being implemented on the PC, and this has meant that the high processing power of the C40 could not be utilised. The processing is thus very slow.

7.4 Summary

The transcription system has made a reasonable attempt at transcription of a wide (but still restricted) variety of pieces. Each piece has highlighted different problems in the transcription process. It has thus demonstrated brittleness, a poor ability to be generalised to other inputs. The suitability of the system for resynthesis is also questionable, as the eventual data format of the timbral and control information is unwieldy, and would not be efficient to synthesise.

There are two main drawbacks in the original design. The first is the fact that whatever our choice of Q, it is *static*. We cannot adapt the encoding to short or long entities. The second is the fact that the system is very block-based, in that the time and frequency must be chosen from a discrete set. We cannot represent things at other times or frequencies. Blocks, even windowed blocks, are inherently discontinuous and create spurious information.

In the next chapter I start redesigning the system, using a smooth and simple wavelet-based representation that does not suffer from either of the above drawbacks.

8. Acoustic Quanta

The previous chapters outlined an analysis/resynthesis scheme based on block-based additive synthesis. In this chapter I examine the hypothesis that Gabor wavelets would be more suitable fundamental building blocks for analysis and resynthesis than the previous multirate STFT blocks. These wavelets are mathematically simple and computationally modest, and appear to be well suited to musical applications.[Nunn 96] I first outline the principles and equations behind quanta. I then give examples of their applications in analysis, transformation, and synthesis, and present preliminary results from implementation on the PC and the C40 platforms. I conclude by comparing the two systems and outlining proposals for future research.

Figure 113 - Dennis Gabor.

8.1 Introduction

A landmark paper by Dennis Gabor in 1947 discussed how quantum theory could be applied to acoustic signals.[Gabor 47, Gabor 46] It explained how any signal may be built up from elementary entities that are characterised by a few parameters. These signals are wavelets consisting of a complex sinusoid multiplied by a Gaussian envelope. Gabor called these wavelets *quanta*, and claimed, "All sound is an integration of grains, of elementary sonic particles, of sonic quanta".

Gabor suggested that a signal could be decomposed into Gabor wavelets at regularly spaced times and frequencies. However, these wavelets are not orthogonal, so an integral transform could not be derived easily. Later, Bastiaans showed that this can indeed be done.[Bastiaans 80, Bastiaans 85] This involves a process similar to the STFT but with Gaussian windows, and is known as the *Gabor transform*.[Daubechies 88b, Daubechies 90, Redding]

The comparatively new field known as wavelet theory is outlined in several papers.[Daubechies 88a, Kronland-Martinet 88, Daubechies 92, Delprat 92, Jawerth, Wilson 92a, Rioul 91, Sweldens 93, Sweldens 96, Graps, Weiss, Edwards, Mallat 89]

It absorbs some of Gabor's ideas, and also adopts some of the multirate techniques described in earlier chapters. Wavelet theory largely uses the wavelet transform (WT), which allows an efficient implementation of a scheme that analyses signals in terms of shifts and dilations of a constant-Q mother wavelet. The choices[Jawerth] include the Haar wavelet, the Coiflet, the Meyer wavelet[Meyer], Daubechies' wavelets[Daubechies 88a], and others. The mother wavelet generally has *finite support* in the time domain in order to allow a fast integral transform. In most cases, the wavelet must be shifted by integers and dilated by factors of two, using a technique known as *voicing*.[Rioul 92, Weiss] It is also possible to use wavelets that are bounded in the frequency domain (and hence infinite in the time domain) – these are called 'harmonic' or 'musical' wavelets[Newland 93, Newland 94], and wavelets with a linear chirp in frequency.[Baraniuk]

There is no reason to use a single wavelet – some methods choose wavelets from several *dictionaries*. Analysis may try to extract the best representation by Matching Pursuit[Mallat 93], Best Orthogonal Basis[Coifman 92], or Basis Pursuit[ChenS 94] procedures. As a signal can be built from wavelets in an infinite number of ways, criteria for determining which set is best must be defined. Possibilities include the *information cost*[Coifman 92], i.e. the number of bits it takes to model a given function to a given accuracy. Other possibilities including entropy and log-energy are discussed by Graps.[Graps]

8.2 Comparison of quanta and wavelets

The wavelets derived by the Gabor transform are defined by shifts and modulations, and are hence characterised by time and frequency. Those derived by the wavelet transform are defined by shifts and dilations, and are characterised by time and *scale*.[77] All derivatives of Gaussians are continuous, whereas the wavelet transform often uses less smooth, or even discontinuous, functions. Non-smooth wavelets, such as Haar wavelets, are well suited to coding discontinuous data such as images but poor for continuous signals.

A signal that is finite in one domain must be infinite in the other, so time-limited wavelets give poor frequency localisation.[Daubechies 88a] Gabor wavelets are infinite in both domains, but in one sense are the most compact – they have a time-bandwidth product of unity. This property gives the advantages of mathematical simplicity and symmetry, but the disadvantage that in principle a wavelet covers the entire time-frequency plane. In addition, the wavelets overlap and thus are not orthogonal.

The Gabor transform is in practice similar to the windowed Fourier Transform, and gives a constant Δf but a widely varying Q. A similar analysis can be done using the multirate techniques described in the previous chapter, allowing a more-or-less-constant Q. The wavelet transform gives a constant Q, in very much the same way as the multirate STFT. In all these cases, Q is *static*. Yet if we are to model music efficiently, it is preferable to allow Q to be chosen to allow longer or shorter basic units, *as appropriate*, to construct the signal. This would allow the efficient coding of music that contains short spectral lines (xylophone, hi-hat[78]) and long spectral lines (didgeridoo, bagpipe drone). Within a single note, it allows accurate coding of the high detail in the attack and compact coding of the slowly changing decay. Dynamically varying Q means that our elemental entities now have three parameters; the length is analogous to the scale parameter of the WT.

---

[77] The (overcomplete) MFT has all three – time, frequency, and scale, as do some wavelet variants.[Coifman 92]

[78] While percussion instruments are often modelled by noise, it must be borne in mind that many do have distinct frequencies of vibration, even when they are not long enough to convey a pitch.

The table below compares many of the representations according to their bandwidth and parameters.

	constant Δf	*pseudo-constant Q*	*several constant Q*	*dynamic Q*
time	PCM			
frequency	FT			
time/frequency	Short-Time FT Gabor Transform	Multirate FT BQFT	Multirate Multiresolution FT[79]	(enveloped ASWS)
time/scale		Wavelet Transform		
time/frequency/scale			Multiresolution FT	**IDEAL TRANSFORM**

Table 39 - Comparison of various transforms.

With integral transforms, the parameters are restricted to those falling on a predefined rectangular grid, a finite and fixed set of times, widths and frequencies. However, a constant 440-Hz tone should ideally be represented as a constant 440-Hz entity, rather than by time-varying tones at 430.664 and 452.197 Hz plus other sidelobes at even more wrong frequencies.[80] It is thus preferable to use an unrestricted set of parameters in order that we can minimise the number of entities required. A simple cost function would be the number of quanta required for 16-bit accuracy.

8.3 Applications

Kronland-Martinet discusses several applications of the wavelet transform to music.[Kronland-Martinet 87, Kronland-Martinet 88] Victor Wickerhauser also discusses sound synthesis and compression.[Wickerhauser] Daniel Arfib successfully used Gabor wavelets for time-stretching with 'not too many artefacts'.[Arfib 90, Arfib 91] He used a sliding-window STFT for the analysis, but used a Hanning window rather than a Gaussian. Boyer implemented transformation using the wavelet transform[Boyer], and Ellis discusses time-stretching.[Ellis 92b] Wavelets have also been applied to source separation and denoising[Popovic, Coifman 90, Coifman 92, BergerJ 94a, BergerJ 94b, BergerJ 94c, BergerJ 95, Delprat 92, Wilson 92a] Gribonval uses Gabor wavelets for analysis of piano tones, using Matching Pursuit[Mallat 93]; this is discussed later.[Gribonval, ChenS 94, ChenS 95, ChenS 96] Kussmaul describes applications to the pitch contour[Kussmaul], and several researchers have examined applications to the rhythm.[Todd, Tait]

8.4 Definitions

In Gabor's paper[Gabor 47], the elementary signals, or quanta, are defined as:-

$$e^{-\alpha^2.(t-t_0)^2}.e^{-2.\pi.f_o.t}$$

[79] This corresponds to our previous Multirate FFT but using sizes of 8, 16, 32, 64, ... simultaneously.

[80] This calculation is for a 44100-Hz sample rate and an FFT of size 2048.

Figure 114 shows a typical quantum. Here, t_0 and f_0 (real) are the positions in time and frequency, and m is the complex magnitude.

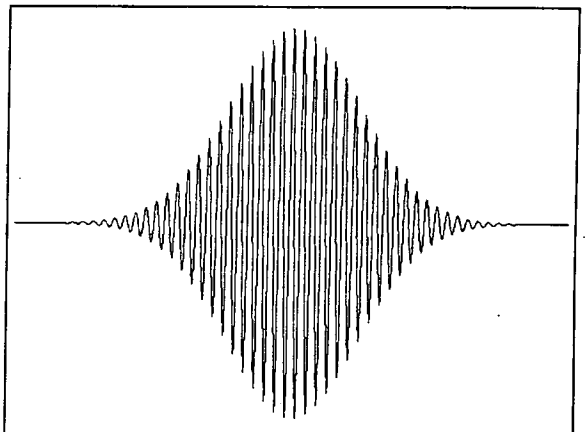


Figure 114 - Waveform of a quantum.

Where Δt_s and Δf_s are the 'inertial' width, the Heisenberg inequality means that $\Delta t_s \times \Delta f_s \geq 1/(4\pi)$, as derived in Appendix G.[Solbach 96b, Papoulis] With Gabor's definitions, Δt and Δf are $2\sqrt{\pi}$ times Δt_s and Δf_s respectively.[Gabor 47] The width in time, Δt, is given by $(\sqrt{\pi})/\alpha$, and the width in frequency, Δf, by $\alpha/(\sqrt{\pi})$. For Gabor wavelets $\Delta t \times \Delta f = 1$, representing the most compact encoding in time-frequency space.[81] This parameter α has the dimensions of Hz.

In this work, I will use 'a' to represent α^2; thus, a has the dimensions of s^{-2} or Hz^2. This parameter will be referred to either as alfa (reducing confusion with Gabor's correctly spelled alpha), or the *density* of the quantum. A high alfa represents a short broadband signal; a low alfa represents a long pure signal. With this convention, $\Delta t = \sqrt{(\pi/a)}$ and $\Delta f = \sqrt{(a/\pi)}$; thus, $a = \pi/\Delta t^2 = \pi \Delta f^2$. For the signal to have finite power, alfa must be positive. If alfa is zero, then the signal has constant magnitude and infinite power. A negative alfa could potentially cause overflow, underflow, divide-by-zero, log-domain, sqrt-domain or malloc failure. In software this situation is trapped by `crash("-ve alfa")`, which starts the shutdown routine of alerting the user, freeing memory, releasing XMS memory, deleting temporary files, de-installing the mouse, switching to text mode, and returning to the start-up directories.

The quanta are in the complex domain – to deal with real signals, we assume either that each quantum of positive frequency is accompanied by one of negative frequency, or that we are dealing with the analytic signal.[82]

8.4.1 Notation

Each quantum has a time, a frequency, an alfa, and a complex magnitude. The notation Q(t,f,a,m) will be used for a quantum.

---

[81] For comparison, the best value for gammatone filters[PattersonR] is $\Delta t \times \Delta f = \sqrt{(5/2)} \approx 1.581$.[Solbach 96b, Wöhrmann]

[82] The analytic signal corresponding to x(t) is x(t)+i.H(x(t)), where H(x(t)) is the Hilbert transform of x(t). The Hilbert transform is equivalent to convolution with π/t.

8.4.2 Typical parameters

Let us attempt to form some typical values for the parameters corresponding to musical notes at various pitches, as shown in Figure 115.

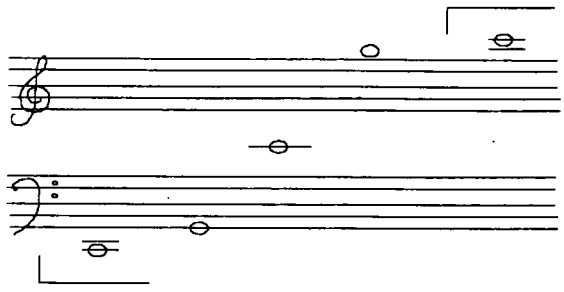


Figure 115 - Very low, low, average, high, and very high frequencies.

The table below gives the expected length in seconds of a note that would be labelled 'short', 'very long', etc., *at that pitch.* (The notes in this table only correspond to the convenient but inaccurate 'scientific tuning' where middle C has a frequency of 256 Hz rather than 261-ish.)

Pitch	Note	f (Hz)	*Very Long*	*Long*	*Average*	*Short*	*Very Short*
Very High	C7	2048	10	3	0.3	0.04	0.02
High	G5	768	15	4	0.4	0.05	0.02
Average	C4	256	20	5	0.5	0.1	0.02
Low	G2	96	30	8	1	0.2	0.05
Very Low	C1	32	30	10	2	0.5	0.2

Table 40 - Length of long and short, and high and low, notes.

Let us suppose that the envelope of a partial is formed using a quantum of this length. For the 'longest' duration of 30 seconds, alfa=0.0035. For the shortest, 20 ms, alfa=7854.

Alfa must conceptually span the range from 0 to ∞; alfa=∞ is quite a common case. In certain situations, such as widening a quantum, alfa may also be negative *if numerical stability can be maintained,* although the corresponding function has infinite power. (Alfa=-∞ is infinitely infinite, meaning $\exp(\infty.x^2)$ or $\delta^{-1}(x)$, and has no obvious use.)

In practice, it is necessary to guard for divide-by-zero errors, and possibly overflow, underflow, or precision loss. Thus, alfa is restricted to the range `AFOREVER` to `ACLICK`, where `ACLICK` is extremely large ($\pi \times 10^{18}$), and `AFOREVER` is `ACLICK`$^{-1}$. These are set just within the maximum and minimum values of a 32-bit `float` in Turbo C. Alfa may also be zero, but these quanta cannot be synthesised.

Thus, the parameters are in these ranges:-

variable	*name*	*units*	*theoretical limits*	*actual limits*
t	time	s	unrestricted	$\pm \pi \times 10^{18}$
f	frequency	s^{-1} or Hz	$-\infty - +\infty$	0 – Nyquist Rate
a	density	s^{-2} or Hz^2	$0, \varepsilon - +\infty$	0, AFOREVER – ACLICK
m	magnitude	none, V, m, Pa	$-\infty - +\infty$	-32767 – +32767

Table 41 - Parameters of a quantum.

8.4.3 Computation

The quanta must ultimately be converted to equally-spaced samples, and this can be done recursively. Nuttall proves that the exponential of a k^{th} order polynomial at equally spaced points requires k complex multiplications.[Nuttall 87] For a second-order polynomial, $x=e^{(c_0+c_1 n+c_2 n^2)}$, with c_0, c_1, and c_2 complex, we need two complex multiplications per sample, which is equivalent to eight real multiplies and four real additions.[Jones 87] This case corresponds to a generalisation of our quanta to include a linear *chirp* in frequency. Since we are (for the time being) assuming no chirp, c_2 is real, and we need six real multiplications and two real additions. If only the real part of the output is required, only four real multiplications and one real addition need be performed for each sample. The round-off error in the recursive computation was shown to be minimal.[Kaiser]

On both the PC and the C40, all parameters are stored as 32-bit `floats`, so one quantum requires 160 bits of memory.

8.5 Basic operations

The simplicity of the mathematical form of the Gabor wavelet allows many straightforward mathematical transformations. Here I outline the basic mathematics used in manipulating quanta.

8.5.1 Identity elements

identity for addition:- Q(any, any, any, 0)

identity for multiplication:- Q(any, 0, 0, 1)

identity for convolution:- Q(0, any, ∞, 1)

The existence of these "don't care" values should be noted; it is meaningless to ask the 'time' of a constant value or the frequency of an impulse.

Note that impulses are a special case – $m.\delta(t)$ is equivalent to $Q(t,0,\infty,m.\infty)$ but is actually denoted by $Q(t,0,\infty,m)$. Impulses are commonly used and must be handled robustly by the low-level routines. Another common case is where a=0, i.e. the quantum is infinitely long. Dannenberg highlights this case as potentially problematic.[Dannenberg 92] However, when used for a finite sound, they will at some stage be multiplied by an envelope that is finite in time.

8.5.2 Negation and inversion

The additive inverse of Q(t,f,a,m) is Q(t,f,a,-m), summing to Q(t,f,a,0). The multiplicative inverse of Q(t,f,a,m) is Q(t, -f, -a, 1/m). Forms with a negative density are not strictly 'quanta', and must be used cautiously as they have infinite energy. The product is Q(undefined, 0, 0, 1).

8.5.3 Fourier Transform

The Fourier transform of Q(t,f,a,m) is $Q(f,t,\pi^2/a,m)$. Note the factor π^2 here – it might be tempting to adopt a *re*definition where a` replaced πa, such that the transform of Q`(t,f,a`,m) is Q`(f,t,1/a`,m), but this has not been used.

8.5.4 Multiplication

If $Q_0=Q(t_0, f_0, a_0, m_0)$ and $Q_1=Q(t_1, f_1, a_1, m_1)$ then $Q_0*Q_1=Q_1*Q_0=Q(t_k, f_k, a_k, m_k)$, where:-

m_k	=	$m_0*m_1*e^{-Z}$	=	$m_0*m_1*Q(t_0-t_1, 0, a_0*a_1/a_k, 1)$
Z	=	$-(a_0*a_1/a_k)*(t_0-t_1)^2$		
a_k	=	a_0+a_1		
f_k	=	f_0+f_1		
t_k	=	$(a_0*t_0+a_1*t_1)/a_k$		

The frequencies and the densities add, but the time is intermediate, weighted by the densities, and the magnitude is reduced according to the time difference. In general, the product will have a larger density, but if $-a_0<a_1<0$, then the product can be wider.

It is possible to approximately move a quantum to a different position by multiplying it by another quantum. It is not *strictly* possible to do a pure time shift from $Q(t_0,?,a_0,?)$ to $Q(t_k,?,a_0,?)$ as keeping alfa the same requires $t_1=\infty$. An approximate shift from $Q(t_0,?,a_0,m_0)$ to $Q(t_k,?,a_0+\varepsilon,m_k)$ requires $t_1\approx a_0(t_k-t_0)/\varepsilon\rightarrow\infty$ and $Z\approx-(a_0^2/\varepsilon)(t_0-t_k)^2$, and we thus risk losing precision. Of course, if we can handle pseudo-infinity robustly, it is possible to perform the time shift this way, but the simpler way is to directly change the times, as discussed in 'Direct operations' later.

8.5.4.1 Exponentiation

If $A=Q(t, f, a, m)$ then $A^2=Q(t, 2f, 2a, m^2)$. More generally, $A^n=Q(t, n*f, n*a, m^n)$, because:-

$$\left(m\cdot e^{-a\cdot t^2}\cdot e^{-2i\cdot f\cdot t}\right)^n = m\cdot e^{-a\cdot n\cdot t^2}\cdot e^{-2i\cdot f\cdot n\cdot t}$$

This allows the creation of strictly harmonic tones with a=0. If q=Q(?, 261, 0, 1), then an *infinite* square wave can be defined as $q+q^3/3+q^5/5+q^7/7+\ldots$. Note that if q had a>0, the duration of A^n would be shorter, so the timbre would be time-dependent.

8.5.5 Convolution

The time convolution of two quanta, denoted $Q_1 \otimes Q_2$, can be determined easily since it corresponds to multiplication in the spectral domain. In fact, we can consider multiplication and convolution to be a single operation with the focus either on the time or the frequency axis.

$Q(t_0, f_0, a_0, m_0) \otimes Q(t_1, f_1, a_1, m_1) = Q(t_1, f_1, a_1, m_1) \otimes Q(t_0, f_0, a_0, m_0) = Q(t_k, f_k, a_k, m_k)$, where:-

$$
\begin{aligned}
t_k &= t_0+t_1 \\
f_k &= (f_0*a_1+f_1*a_0)/(a_0+a_1) \\
a_k &= a_0*a_1/(a_0+a_1) \\
m_k &= m_0*m_1*e^{-Z} \\
Z &= (-\pi^2*(f_0-f_1)^2)/(a_0+a_1)
\end{aligned}
$$

As before, a pure time shift is impossible – convolution would have to be carried out with $a_1=\infty$. However, an approximate shift is of course possible with a_1 very large.

Convolution and multiplication are in practice carried out by the same function – for convolution, we first perform the 'instant Fourier transform' by turning the arguments *inside-out* (by swapping their time and frequency arrays and replacing each alfa by π^2/alfa), then use the multiply function, and similarly turn the result inside-out.

Musical applications of convolution, including delays, filtering, cross-synthesis, and rhythm, have been discussed by Roads and others.[Roads 92, Roads 93, Roads 94] Roads primarily discusses convolution of the waveform – this is a computationally expensive step and in most cases convolution is instead carried out by *fast convolution*, based on multiplication of FFTs.[Stockham 69] We are convolving the tokenised quanta symbolically rather than convolving the waveforms themselves, but the results are identical.

The counterpart of exponentiation for convolution will be termed *convolentiation* until a better or accepted term is found. If $A=Q(t,f,a,m)$ then $A^{<n>}=Q(nt,f,a/n,m^n)$ where $A^{<5>}=A\otimes A\otimes A\otimes A\otimes A$.

8.5.6 Addition

The sum of two quanta cannot in general be expressed as a single quantum, unless they share the same time, frequency, and density.

8.5.7 Sequences

If we form a sum of quanta $Q(\{...,-2,-1,0,1,2,3,...\},0,a,m)$, and the quanta are sufficiently broad and close, the sum will be close to constant. If the quanta are broad and alternate in sign, i.e. $Q(\{...,-2,-1,0,1,2,3,...\}, 0, a,\{...,m,-m,m,-m,m,-m,...\})$, then we can form a good approximation to a sinusoid. The fact that this is an approximation does not necessitate lossiness, as the error term can be computed as quanta too. If the quanta are regularly spaced but do not overlap sufficiently, then we can form an expression relating the periodic DC quanta to a set of quanta with frequencies at multiples of the repetition frequency.

Since we can form a set of DC quanta sum to (approximately) unity, we can argue that another quantum equals itself times unity, which equals itself multiplied by the set of quanta. This gives an expression allowing us to break any quantum into overlapping entities, and more importantly, a method of combining quanta into longer and fewer quanta.

8.5.8 Stereo

So far we have ignored the issue of stereo. There are two possible ways to extend the above to stereo samples. We could assign all quanta to either left or right, or give the quanta themselves another parameter corresponding to spatial position. With the second method, the parameter p of a quantum means that the quantum becomes:-

$$\text{RIGHT} = \text{Re}(\exp(-a*t^2 + 2\pi it)) * P(p)$$
$$\text{LEFT} = \text{Re}(\exp(-a*t^2 + 2\pi it)) * P(-p)$$

We still have to define P(p) such that quantum operations are mathematically regular. Left to right could be mapped onto -1 to +1 or $-\infty$ to $+\infty$. In the former case, $P(p) = 0.5*(1+p)$ gives a linear slope, which results in an uneven pan. Better would be $P(p) = 0.5*(1+\sin(\pi*p/2))$. In the latter case, $P(p)=\exp(-ap*(p\pm0.5)^2)$ gives Gaussians centred at the left and right speakers. In Roads's paper on convolution, he points out that stereo placement is essentially the same as convolving the sound with the impulse responses at the two speakers.[Roads 93] These responses, known as head-related transfer functions (HRTF), are detailed by Kendall.[Kendall]

Adding a parameter such as position means extending the model to 32 species, but with the compact representations, this is a relatively small increase. Very often we would design an instrument in mono as a species-15 molecule, then place it in a fixed position by what would be species-16. This extension to stereo placement has not yet been implemented, but it appears that this is a relatively simple process. Quadraphonic sound is also feasible.

8.6 Higher-level structures

In most cases it is expected that quanta will be grouped together into higher-level entities. In forming groups of quanta, we anticipate that they might have one or more parameters in common. For example, we might wish to specify a rhythm by a group of quanta that have different times but the same frequency, density, and magnitude. A chord would be a group of quanta with the same times and densities but different frequencies and magnitudes. In order to minimise memory usage, this was made a fundamental consideration in designing higher-level structures for quanta.

The entities are not as complex as the *events* used in granular synthesis by Roads[Roads 88], which are characterised by not only time and duration, but also by waveform, frequency, bandwidth, grain

density[83], and amplitude and the slopes of these five quantities. Truax's *tendency masks* are similar entities.[Truax 88] However, these would be a relatively straightforward extension to implement.

8.6.1 Atoms

The second-lowest unit is called an *atom*. It represents an arbitrary number of quanta, and its *species* determines the topology of the arrays. There are sixteen species, corresponding to whether there are multiple times, frequencies, densities, and/or magnitudes. A species-0 atom is a single quantum. Species 8 has multiple times but only one frequency, density, and magnitude, and could describe a rhythm. A chord could be a group of quanta with the same times and densities but different frequencies and magnitudes, which is species 5.

A word on terminology is in order. While there may well be interesting analogies between acoustic quanta and quantum physics, the term 'atom' is chosen purely out of the need for a term, rather than any direct physical analogy. Note also that some authors, including Gabor, use the term 'atom' for the quanta themselves.

The notation for quanta is extended using braces such that, for example, $Q(\{t_0,t_1,t_2,t_3\},f,a,m)$ represents an atom of four quanta with different times but the same f, a, and m.

Species	*Times*	*Freqs*	*Alfas*	*Mags*	*Examples*
0	1	1	1	1	single quantum, crude filter, delay
(1)	1	1	1	N	(sum to species 0)
2	1	1	N	1	?
3	1	1	N	N	symmetrical shape
4	1	N	1	1	temperament, chord, scale, vowel
5	1	N	1	N	steady-state timbre, tremolo, general filter
6	1	N	N	1	?
7	1	N	N	N	symmetrical tone
8	N	1	1	1	spikes, barlines, martellato
9	N	1	1	N	simple control, reverb, resonance
10	N	1	N	1	rhythm, envelope, partial
11	N	1	N	N	weighted rhythms
12	N	N	1	1	melody profile
13	N	N	1	N	weighted melody profile
14	N	N	N	1	melody
15	N	N	N	N	tune, MIDI, note

Table 42 - Species of atoms.

[83] Grain density is the number of grains to be generated over the duration, and is not connected to the density of a quantum as I have defined it.

The advantage of these types is clear – species 2-14 are more compact than 0 or 15, and can be manipulated and synthesised faster. I will refer to these species as *compact*.

Note that when N=1 (or N=0), the species is irrelevant and the atom can be validly viewed as any species. Figure 116 schematically shows the sixteen types arranged in a Karnaugh map.[84]

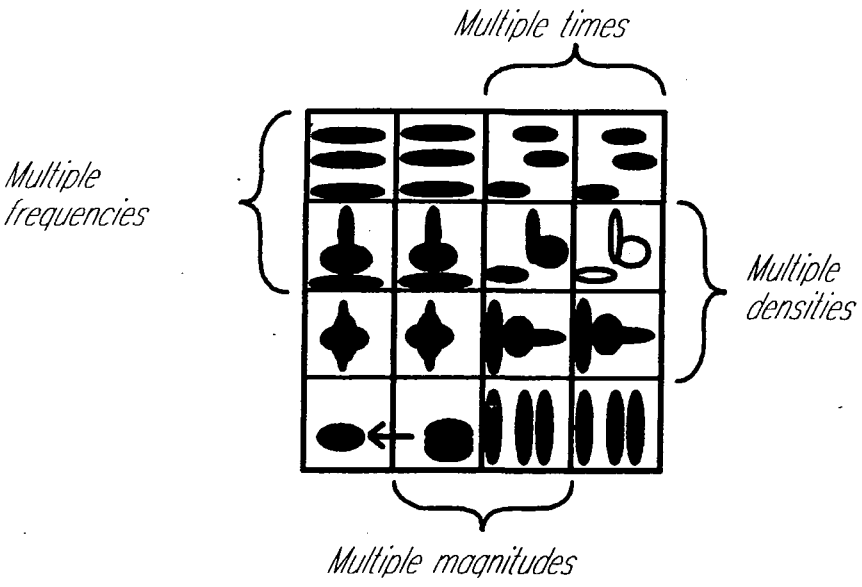


Figure 116 - Karnaugh map illustrating the sixteen species.

8.6.2 Atom operations

8.6.2.1 Direct array operations

Many operations can be carried out by manipulating the four arrays separately. These operations are referred to as *direct* operations, and may be non-linear. A simple example is time displacement, which, as we saw earlier, required the use of pseudo-infinity when calculated by either multiplication or convolution. It is much simpler to add the delay time to the time array and leave the other three alone. This is implemented by `Dadd(dt,0,0,0,0)` – Directly ADD to the four arrays dt, 0, 0, and (0,0) – the operations +0 and *1 are trapped and not carried out.

8.6.2.2 Addition

Atoms can be added by concatenating the arrays, as long as both are first promoted to a compatible species. For example, adding an a tom of species 3 to one of species 9 means converting both to species (species1 OR species2) = 11. The sum may be another species still – in this case adding Q({N_1 times}, f_1, {N_1 alfas}, {N_1 mags}) to Q({N_2 times}, f_2, {N_2 alfas}, {N_2 mags}) must be converted to species 15 unless $f_1=f_2$. (Alternatively, atoms of different species can be joined into a *molecule* – see later for details.)

---

[84] A Karnaugh map is an arrangement of 2^N values according to the bits in their index. A change in one bit moves to an adjacent cell. It is typically used for designing logic circuits.

8.6.2.3 Multiplication and convolution

When two atoms are multiplied, the resultant species is given by the following table.

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	0	0	11	11	4	5	15	15	9	9	11	11	13	13	15	15
1		0	11	11	4	5	15	15	9	9	11	11	13	13	15	15
2			11	11	15	15	15	15	11	11	11	11	15	15	15	15
3				11	15	15	15	15	11	11	11	11	15	15	15	15
4					4	5	15	15	13	13	15	15	13	13	15	15
5						5	15	15	13	13	15	15	13	13	15	15
6							15	15	15	15	15	15	15	15	15	15
7								15	15	15	15	15	15	15	15	15
8									9	9	11	11	13	13	15	15
9										9	11	11	13	13	15	15
10											11	11	15	15	15	15
11												11	15	15	15	15
12													13	13	15	15
13														13	15	15
14															15	15
15																15

*Table 43 - Species of A*B.*

An easier way to express this is:-

many times => many times, many mags
many freqs => many freqs
many alfas => many times, many alfas, many mags
many mags => many mags

When two atoms are *convolved*, the rules for time and frequency are reversed:-

many times => many times
many freqs => many freqs, many mags
many alfas => many freqs, many alfas, many mags
many mags => many mags

If there are N_0 and N_1 quanta in the arguments of multiplication or convolution then the result will have $N_0 * N_1$ quanta.

Note that the combination of atoms of compact species is not necessarily compact. For instance, species 0 * species 6 is the product of a single quantum with an atom that has 1 time, N frequencies, N alfas, and 1 magnitude. However the product also has many times and many magnitudes and thus has species 15. The rules above show that the most problems arise when the quanta have different alfas. However, in many other cases the combination does lead to a case where only the required arrays are computed, benefiting from the adoption of compact representations.

8.6.3 Transformations

Many common audio transformations can be expressed compactly as simple linear operations between groups of quanta. The range of musical transformations that can be implemented easily is best illustrated by some examples.

If one atom holds a weighted set of frequencies Q(0, $\{f_0,f_1,f_2,...\}$, 0, $\{m_0,m_1,m_2,...\}$), we can apply a control envelope to them simple by multiplying the atom by another corresponding to the control envelope.

Since these operations are carried out with the 'tokenised' quanta rather than the actual audio, complex effects can be specified simply, although the resultant number of quanta may be large. It is worth noting, though, that a large number of quanta is not as inefficient as it may seems – short and weak quanta are less costly to compute.

The synthesis technique, which has been compared closely to granular synthesis, seems to incorporate many other techniques. It allows the analog-inspired technique of filtering a fixed waveshape. It improves upon traditional additive synthesis by removing the need for line-segment approximation. It allows subtractive synthesis as a random collection of quanta can approximate noise. AM and hence tremolo can be computed easily by multiplication. It should also allow FM timbres and vibrato as Bessel functions can also be modelled as sums of quanta. It might even be possible to rewrite physical modelling equations in terms of operations between quanta.

8.6.4 Time shifting

Gabor wavelets also allow non-linear editing operations. Arfib used Gabor wavelets to successfully carry out time-stretching, a seemingly simple task but difficult in practice[Arfib 90, Arfib 91] To time-shift a set of quanta, we multiply the times by a ratio, multiply the phases by the same ratio, then reduce the densities by the square of this factor.

8.6.5 Examples

Below I give examples of the sixteen species and discuss which musical concepts they might be suited to.

8.6.5.1 Species 0 – 1111

The notation '1111' above refers to the number of Times, Frequencies, Alfas, and Magnitudes. (Note that magnitudes are complex, and it is not currently possible to have N real values but only 1 imaginary value, or vice versa.)

8.6.5.1.1 Single quantum

Figure 117 shows a feasible representation of a quantum. The axes are time and frequency (conventionally left-to-right and up=high). The magnitude could be shown by colour or shading density. Phase could be shown by an arrow.

(Quanta are often alternatively represented by a rectangle in the t-f plane.)

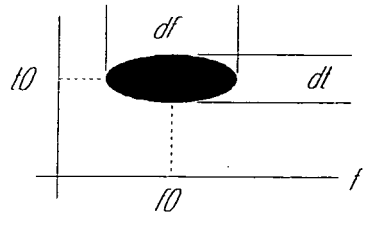


Figure 117 - Single quantum.

8.6.5.1.2 Simplest low-pass filter

A crude filter can be made with a single quantum – Q(0, 0, massive, 1) – with t=0, f=0, df=40000 Hz and thus alfa=π*1600000000 $\approx 5*10^9$ and dt=25 μs. If the input is expressed as a series of quanta, then its convolution with this brief spike will be a series of very slightly broader quanta representing the low-pass filtered version. Note that the frequencies are very slightly distorted by this operation, as indicated by the equations above; since the 'flat' side of f is amplified by more than the 'sharp' side, the centre frequency is effectively lowered.[85] Roads also discusses FIR filtering by convolution.[Roads 93]

8.6.5.1.3 Pure delay

If we have formed a set of quanta corresponding to a note, then the convolution with Q(10, 0, ∞, 1) gives the same note delayed by 10 seconds.

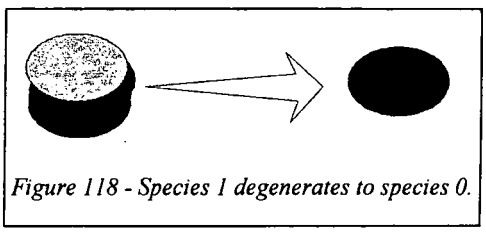


Figure 118 - Species 1 degenerates to species 0.

8.6.5.2 Species 1 – 111N

8.6.5.2.1 Degenerates to single quantum

As shown in Figure 118, this has 1 time, 1 frequency, 1 alfa, and N magnitudes. An atom of species 1 can be converted to one of species 0 by summing the magnitudes. Note that this is the only *degenerate* species of the 16.

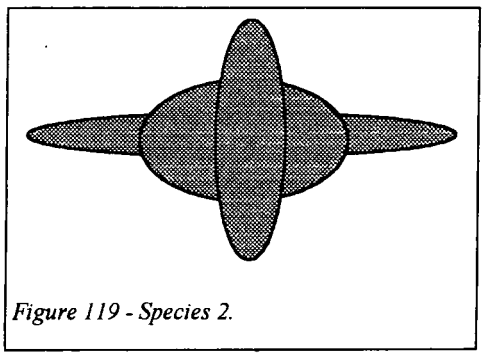


Figure 119 - Species 2.

8.6.5.3 Species 2 – 11N1

This species seems useless – many widths but at the same frequency, time, and magnitude, as shown in Figure 119.

---

[85] The corollary of this effect in the time domain is also interesting. If a set of equally spaced quanta (with identical densities) is multiplied by a control envelope that is increasing in volume, the new quanta will have centres that are later. If the control envelope is a single quantum, then the interval between the new quanta will be a shorter constant. This result can be stated in a somewhat surprising manner: perceptual time (as defined by this interval) appears to go slightly faster while we move a volume slider up and down in a parabolic path.

8.6.5.4 Species 3 – 11NN

Species 3 atoms, shown in Figure 120, have multiple densities and magnitudes but only one time and frequency.

8.6.5.4.1 Symmetrical shape

A species 3 atom is a weighted group of widths and thus can denote a shape symmetrical about the only time, as shown in Figure 121.

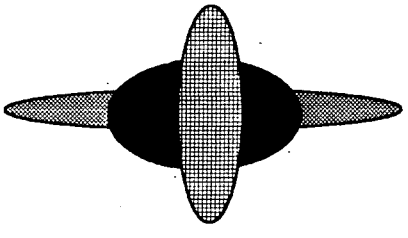


Figure 120 - Species 3.

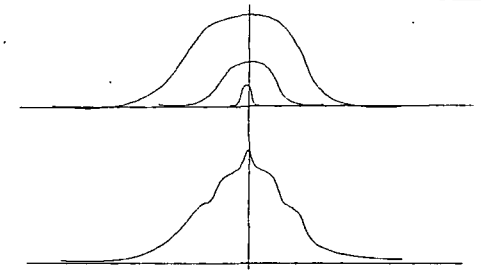


Figure 121 - Symmetrical shape.

8.6.5.5 Species 4 – 1N11

A species 4 atom has many frequencies.

8.6.5.5.1 Temperament/chord/scale

A temperament, chord, or scale could be specified as Q(0, {set}, 0, 1) – a time/duration/amp-independent set of frequencies, as in Figure 122.

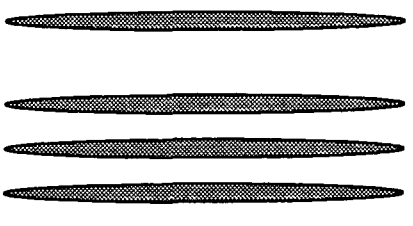


Figure 122 - Chord shape.

8.6.5.5.2 Crude vowel sound

A simplistic definition of the two formant frequencies in the sound 'OO' could be an atom Q(any, $\{f_1,f_2\}$, 0, m). However, species 5 is preferable as it allows weighting of the formants.

8.6.5.5.3 Shephard tones

The paradoxical phenomenon of Shephard tones[Shephard, Risset 91] can be formed by multiplying impulses regularly spaced in the *log*-frequency domain by the spectral envelope desired. While a non-linear operation, it is straightforward to carry out logarithms and exponentials.

8.6.5.6 Species 5 – 1N1N

8.6.5.6.1 Frequency-independent waveforms

The timbre of a square wave can be represented by an atom with 1 time, N frequencies, 1 alfa, and N magnitudes – species 5. In theory we need an infinite number of terms. This is shown in Figure 123, except that alfa in practice would be zero.

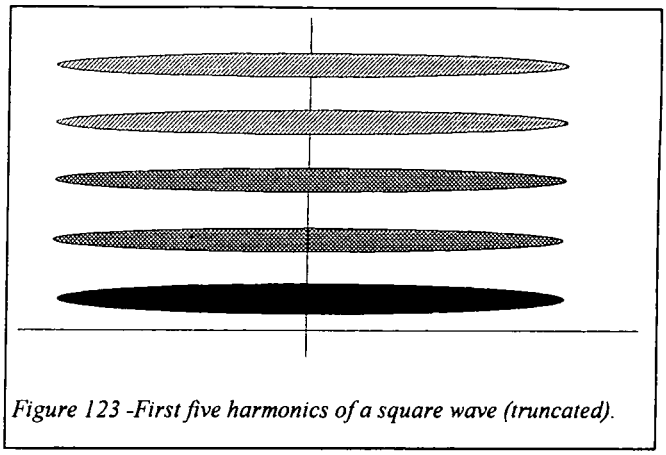


Figure 123 -First five harmonics of a square wave (truncated).

8.6.5.6.2 Amplitude modulation

Conventional amplitude modulation of a signal can be expressed as multiplication of the input quanta by a steady-state gain and modulation parameters. This corresponds to multiplying the input quanta by a species 5 atom Q(0, {0,100}, 0, {1, fraction}).

8.6.5.6.3 Tremolo

A 6-Hz tremolo is a simple case of amplitude modulation – it can be represented as the sum of two quanta – Q(0, {0,6}, 0, {1, 0.1}). This is illustrated *schematically* in Figure 124 – note that in fact dt=∞ and df=0. Multiplying a note by this atom gives the tremolo-ed wave.

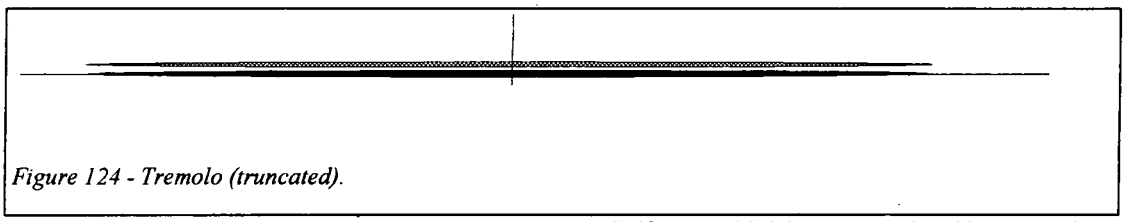


Figure 124 - Tremolo (truncated).

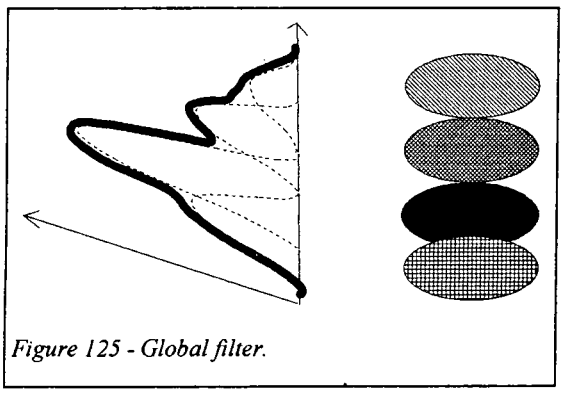


Figure 125 - Global filter.

8.6.5.6.4 General global filter

Global equalisation can be achieved as with species 0 above, by convolving the input with a species 5 atom with a very large alfa, as shown in Figure 125. Here the bandwidths of each 'pass-band' are constant.

8.6.5.7 Species 6 – 1NN1

I have not yet found a use for this species, shown in Figure 126. It is similar to species 2.

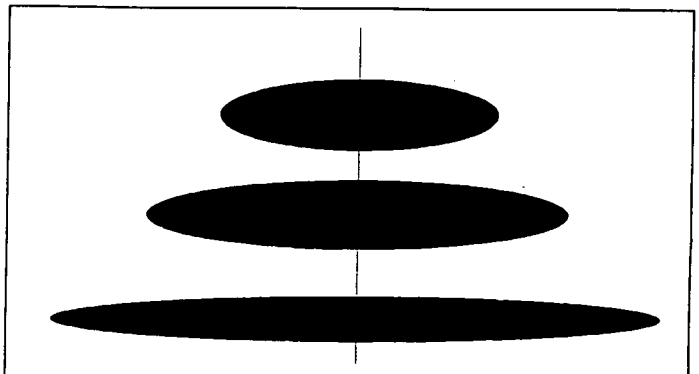


Figure 126 - Species 6.

8.6.5.8 Species 7 – 1NNN

8.6.5.8.1 Symmetrical tone

This is as species 3 but each quantum has its own frequency. It is shown in Figure 127.

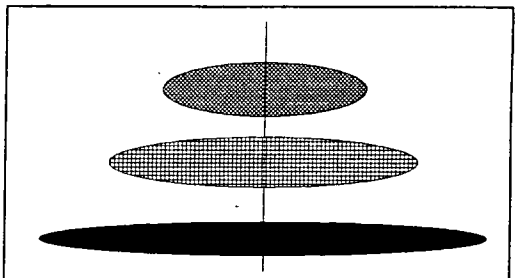


Figure 127 - Symmetrical tone.

8.6.5.9 Species 8 – N111

8.6.5.9.1 Basic rhythm

An unaccented rhythm can be represented by an atom of species 8 with a=∞, i.e. Q({set}, 0, ∞, 1), shown in Figure 128. In this case f is usually zero.

8.6.5.9.2 Metrical structures

The example above can be built up from smaller atoms, also of species 8, as shown below.[86] (This example was also used by Tanguiane[Tanguiane 91], who derived a system to derive the rhythm from the raw onset times by minimising the data requirements.)

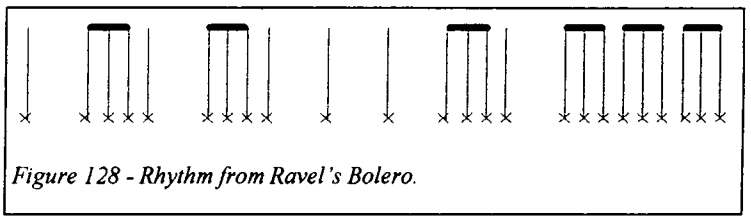


Figure 128 - Rhythm from Ravel's Bolero.

```
crotchet = MakeFillAtom(1,1,1,1,  0,0,∞,1)
triplets = MFA(3,1,1,1, 0,.33,.66, 0,∞,1)          /* abbreviation of MakeFillAtom */
repeat1beatlater = MFA(2,1,1,1,   0,1,  0,∞,1)
repeat2beatslater = MFA(2,1,1,1,   0,2,  0,∞,1)
repeat6beatslater = MFA(2,1,1,1,   0,6,  0,∞,1)
repeat200 = MFA(200,1,1,1,  0,12,24,36,48,...2400,  0,∞,1)
crotcrot = Convolve(crotchet,repeat1beatlater)
crottrip = Join(crotchet,Delay(triplets,1))
triptrip = Convolve(triplets,repeat1beatlater)
mostofbolero = Convolve(Convolve(crottrip,repeat2beatslater),repeat6beatslater)
bolerosidedrumpattern = Join(mostofbolero,Delay(crotcrot,4),Delay(triptrip,10))
bolerosidedrumscore = Join(Convolve(repeat200, bolerosidedrumpattern), sidedrumending)
sidedrumoutput = Convolve(bolerosidedrumscore,userdefinedsidedrumpatch)
```

[86] As the MakeFillAtom function takes a variable number of arguments, a decimal point must be put after integral arguments to ensure they are passed correctly. This is not shown above for simplicity.

8.6.5.9.3 Repetition structure

For a set with larger spacing in time, a species 8 atom with f=0 and a=∞ can mean a structural unit such as the times of bars, verses, sections, and so on.

8.6.5.9.4 Repeated pitch – martellato

f need not be zero, and a species 8 atom could use this. Such an atom could mean '4 semiquavers on high F#'. They would necessarily have the same amplitude.

8.6.5.9.5 Shephard rhythms

Whereas Shephard tones have an incessantly rising time but a constant pitch height, Risset demonstrates what we might call a Shephard rhythm, which incessantly slows down while adding faster beats.[Risset 91] These too can be made using a set of impulses regularly spaced in the *log*-time domain.

8.6.5.10 Species 9 – N11N

In species 9 atoms, there are N times and N magnitudes.

8.6.5.10.1 Accented rhythm

Here, a=∞ and f=0, as shown in Figure 129. This is similar to the above but each atom has a magnitude.

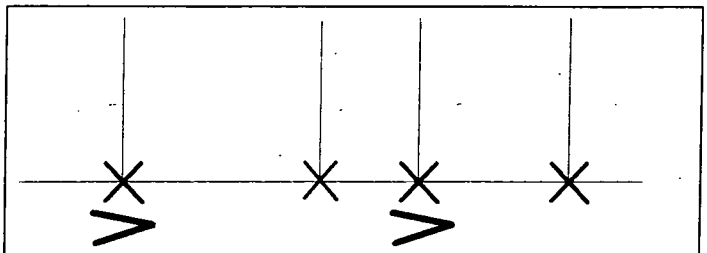


Figure 129 - Accented rhythm.

8.6.5.10.2 Simple control envelope

For a smaller a, we can form a simple continuous envelope, but since a is fixed, it cannot be arbitrarily complex, i.e. it is band-limited by a, as shown in Figure 130. Typically f=0.

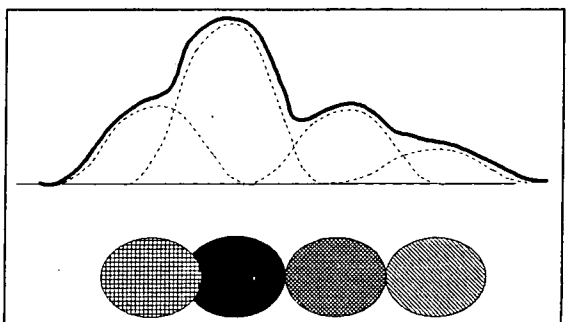


Figure 130 - Band-limited control envelope.

8.6.5.10.3 Echoes and reverberation

Echoes and reverberation can be applied to the quanta representing a signal by convolving them with the impulse response. Convolution with Q({0,0.1,0.2,0.3},0,∞,{1,0.3,0.1,0.03}) gives a simple echo. The sharpness of the echo is determined by the density, which should be infinite to avoid frequency distortion, as shown in Figure 131.

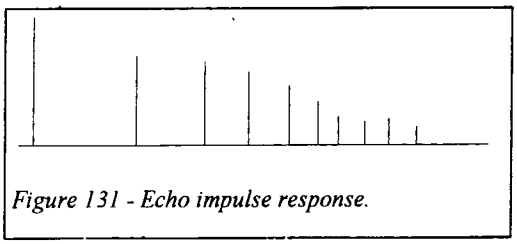


Figure 131 - Echo impulse response.

Reverberation is the same as echoes except the sound is diffuse. This can be modelled as above using a large but *finite* density, i.e. Q({set}, 0, large, {set}). This reverb will be softer, but there will also be frequency colouration.

8.6.5.10.4 Coloured Reverb/Resonance

If the frequency is also non-zero – Q({set}, f, large, {set}) – then we can express a single resonance as a set of quanta with different times and magnitudes.

8.6.5.10.5 Reverse echo and negative delay

In non-real-time situations, it is just as easy to create non-causal effects, achieved by convolution with a quantum at t<0. 'Negative delay' units can also be designed.

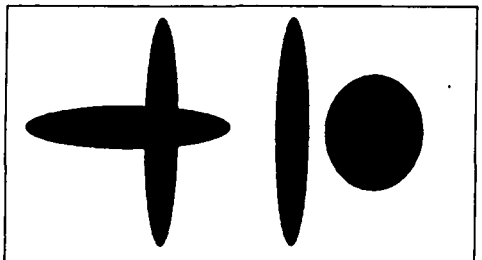


Figure 132 - Species 10.

8.6.5.11 Species 10 – N1N1

A species 10 atom, shown in Figure 132, has N times, one frequency, N densities, and one magnitude.

8.6.5.11.1 Basic rhythms

This differs from species 8 in that each quantum has a different density, as shown in Figure 133. This could be *interpreted* as a rhythm, allowing for the fact that we are reinterpreting {centre time, width} as {start time, duration}. Such a transformation is easy to implement but is non-linear.

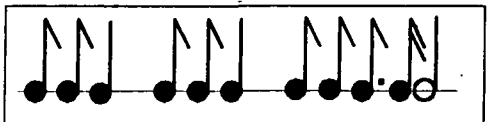


Figure 133 - Basic rhythm of "Jingle Bells".

8.6.5.12 Species 11 – N1NN

Species 11 has many quanta with the same frequency, as in Figure 134.

8.6.5.12.1 Accented rhythm

This could represent an accented rhythm at a certain pitch. as shown in Figure 135.

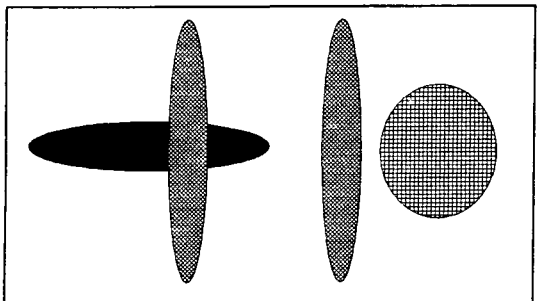


Figure 134 - Species 11.

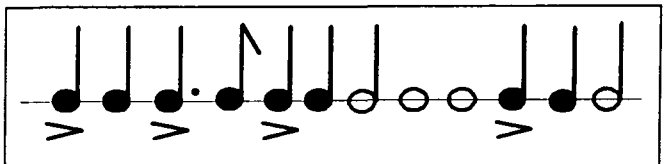


Figure 135 - Accented rhythm, taken from the bass line of the Mendelssohn.

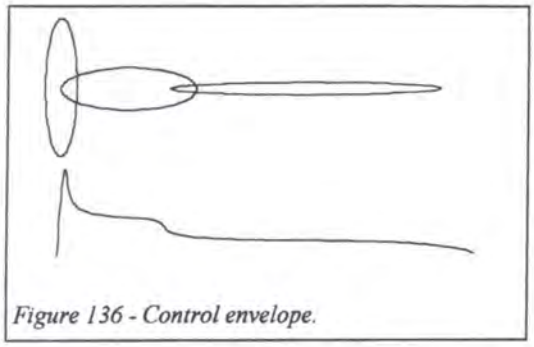


Figure 136 - Control envelope.

8.6.5.12.2 Control envelope

This is a smooth envelope at a fixed frequency, as shown in Figure 136. At f=0 we have a master volume control.

8.6.5.12.3 Single harmonic

Species 11, Q({set}, f, {set}, {set}), is ideal for a single partial of a constant frequency.

8.6.5.12.4 Waveform

During the steady-state portion of a note, the timbre can be described by a single period. In much earlier research by this author[Nunn 84], the steady-state spectra of brass notes were studied. Waveforms and spectra of a tenor trombone at Bb1, Bb2, Bb3, and Bb4 are shown in Figure 137.

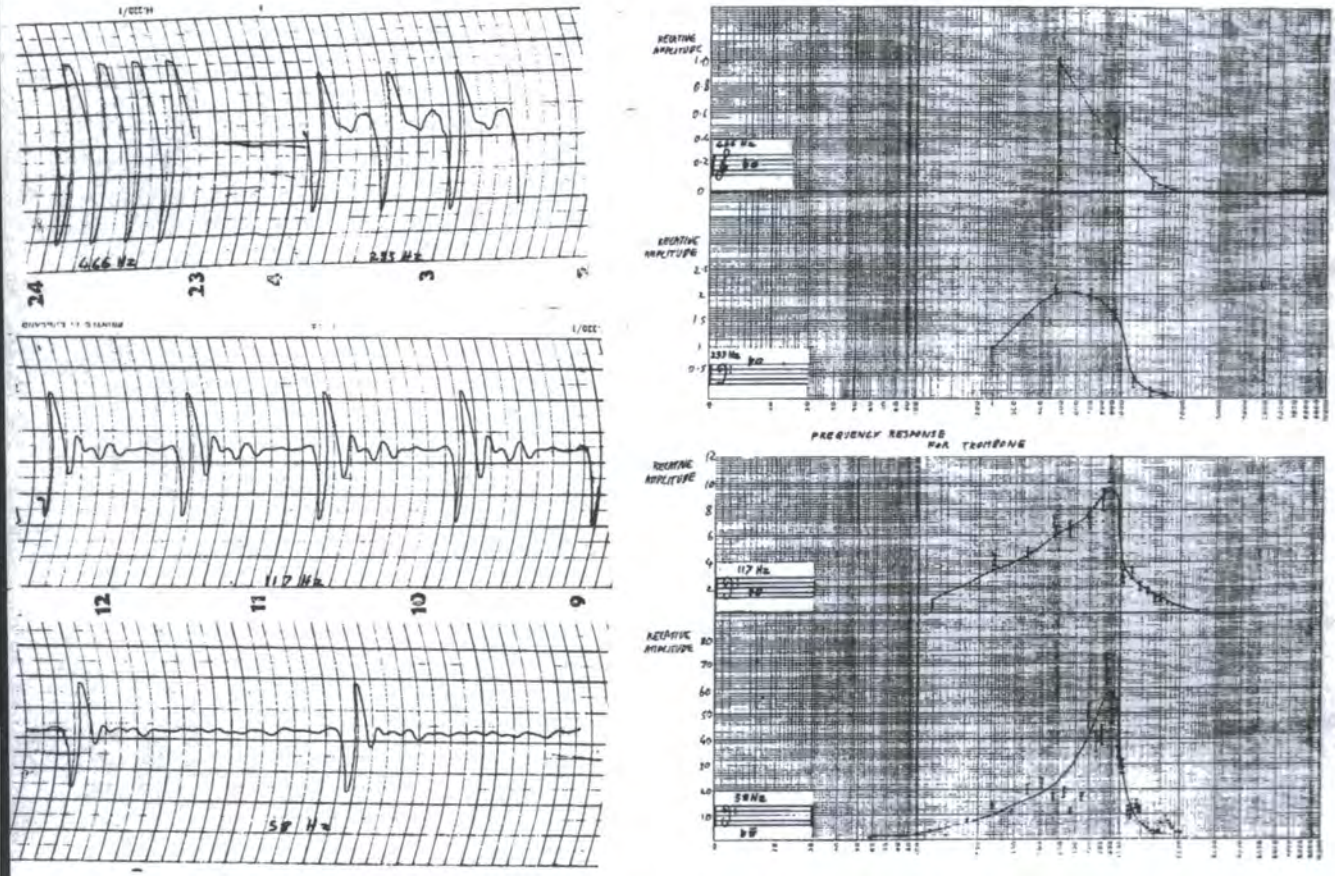


Figure 137 - Waveforms and spectra of trombone notes.

A single period can be converted to a quantum representation by matching Gaussians to the waveform. The sequences of Gaussians can be converted to a spectrum very easily using the sequence relations described earlier. Each 'bump' will repeat at the pitch period, and we can convert these sequences of bumps into the corresponding steady-state spectrum.

If one waveform of the trombone pedal note[87] (58 Hz) were very simply modelled as a species-11 atom of two quanta,

$$Q(\{-0.0005,+0.0005\},0;\{Adur(0.001),\ Adur(0.0012),\{0.3,0.25\})^{88}$$

we convolve this with impulses corresponding to the pitch,

$$Q(\{\ldots,{}^{-2}/_{58},{}^{-1}/_{58},0,{}^{1}/_{58},{}^{2}/_{58},{}^{3}/_{58},\ldots\},0,\infty,1\}$$

and get a species-5 atom representing its timbre.

$$Q(any,\{0,58,2\times58,3\times58,4\times58,\ldots\},0,\{?,1,3.7,7.6,13.7,\ldots\})$$

It will be convenient to implement the Shah function, which is an infinite sum of impulses.

The quantum representation thus permits, and even simplifies, pitch-synchronous granular synthesis – to form a note, we simply repeat this waveform at the pitch period. (As with PSGS, the waveform is pitch-dependent unless it is suitably scaled.)

8.6.5.12.5 Quantum-to-on/off

To convert a quantum with a time of t_0 to a 'note' with a start time of t_0 and an end time of t0+dur, we replace it by a set of quanta summing to a rectangular pulse.

8.6.5.13 Species 12 – NN11

In species 12, shown in Figure 138, all the quanta share a magnitude and an alfa.

8.6.5.13.1 Melody shape

An example is shown in Figure 139; it represents a basic melody with no variation in magnitude.

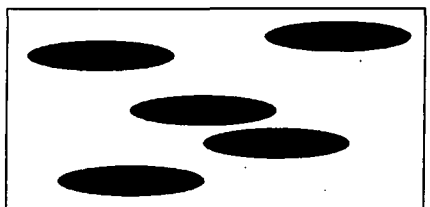


Figure 138 - Species 12.

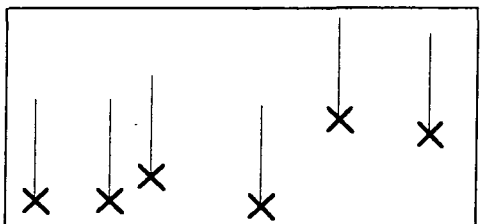


Figure 139 - Melody shape for "Happy Birthday to You".

[87] A pedal note is one played using the lowest mode of vibration of a brass instrument.

[88] These values are very approximate.

8.6.5.14 Species 13 – NN1N

Species 13 has only one density, and is illustrated in Figure 140.

8.6.5.14.1 Weighted melody profile

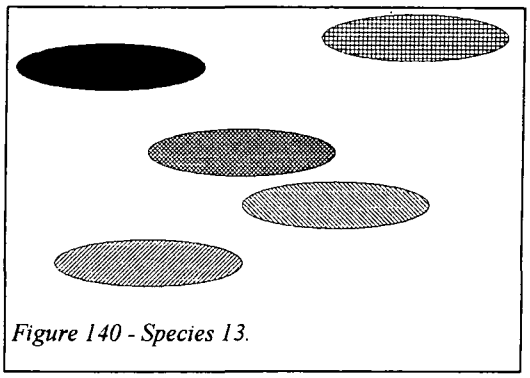


Figure 140 - Species 13.

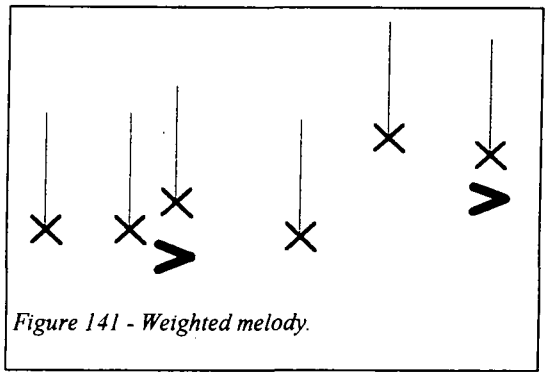


Figure 141 - Weighted melody.

This species can represent a melodic line, as in Figure 141, although the quanta have the same density.

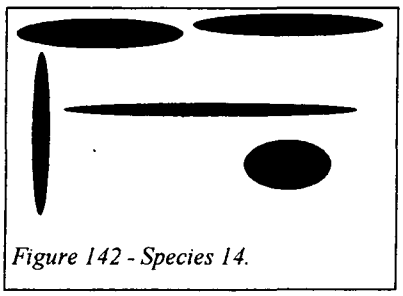


Figure 142 - Species 14.

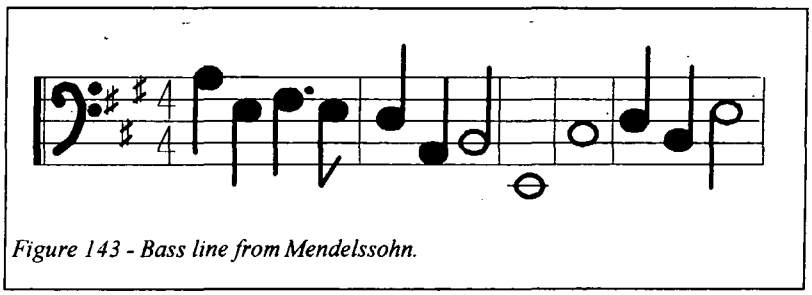


Figure 143 - Bass line from Mendelssohn.

8.6.5.15 Species 14 – NNN1

Figure 142 shows species 14, in which there is only one magnitude.

8.6.5.15.1 Unweighted melody

This is general except there is no amplitude modulation, as shown in Figure 143. It can be thought of as melodies played on a non-velocity-sensitive MIDI keyboard.

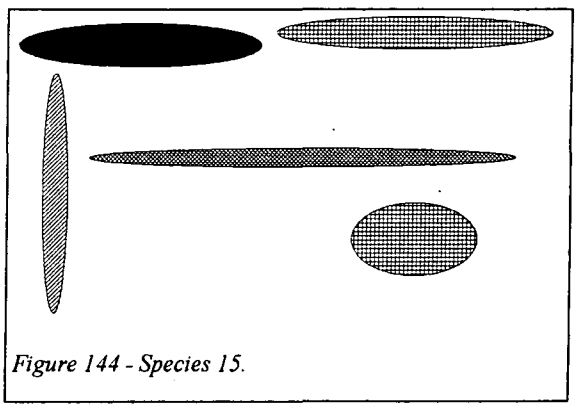


Figure 144 - Species 15.

8.6.5.16 Species 15 – NNNN

Species 15 represents an atom with a full complement of times, frequencies, densities, and magnitudes, as shown in Figure 144.

8.6.5.16.1 MIDI atoms

The notes in a MIDI file can easily be 'converted to' an atom of species 15.

A note with time MT, pitch MP, duration MD, and velocity MM can be represented by Q(MT+MD/2, freq2pitch(MP), π/MD^2, MM). Note that as the Gaussian is symmetrical, the time must fall in the centre of the note.

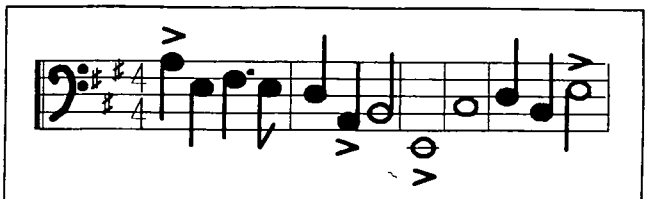


Figure 145 - Weighted notes, as in Mendelssohn bass.

8.6.5.16.2 Timbre

Species 15 can represent an entire timbre. The example in Figure 146 could be a brass-like timbre where the upper harmonics start later and die down sooner.[Benade] As most notes have a complex attack and a simpler delay, the attack has shorter quanta (with Δf larger), and is followed by progressively longer quanta. However for an instrument with a rapid cutoff or offset noise, such as the organ or the harpsichord, the end would also need several short quanta to create the final click.

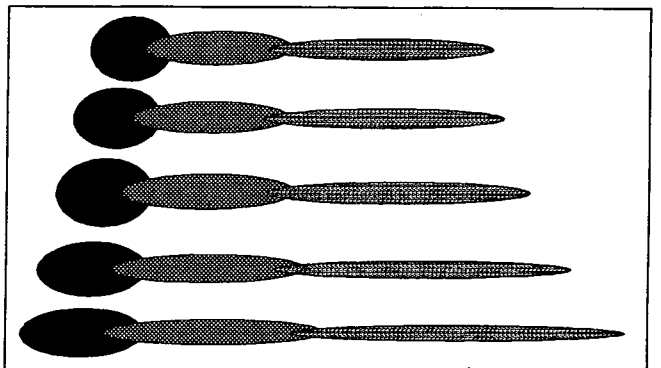


Figure 146 - Simple approximation to brass timbre.

This timbre could not be transposed without causing timbral alteration. In the earlier study[Nunn 84] brass instruments were found to display formant-like behaviour. (Although Moorer claims they do not.[Moorer 76, Moorer 77a]) Slawson cites research showing fixed formant for violins and double-reed woodwind instruments.[Slawson]

A fixed formant, or a set of formants, can be created easily as shown above, and applied after forming and transposing the timbre. Thus, we can at least partly model the dependence of timbre on pitch for many instruments by this approach of combining a fixed waveshape with a fixed filter. This 'excitation & resonator' approach is also widely used for strings, brass[Backus], and the human voice.[Moorer 77a, MooreF 90]

Earlier I described how timbres could be formed from waveforms. Given a note played in several ranges, it may be possible to determine how these could be expressed as fixed-waveshape and fixed-filter components.[89] Alternatively, we could simply interpolate between the sets of quanta.

We can thus derive much of the spectral variation of an instrument by examining the spectra at several times, frequencies, and loudnesses.

---

[89] Other approaches to timbral interpolation include interpolating amplitude envelopes[Horner 95b] and wavetables[Horner 95b] by genetic algorithms.

8.6.5.16.3 Attack noise

Inharmonic noise during an attack, such as created by a hammer, a plectrum, or the throat of a mouthpiece can be modelled by a set of quanta with short durations at t slightly larger than 0. These must be added – there is no linear operation for this.

8.6.6 Molecules

The third-lowest unit is a *molecule,* which is a group of atoms, possibly of different species. The addition, multiplication, and convolution of molecules is simply defined as the sum of the result of the operations on their atoms.

8.6.7 Note on the Fourier Transforms

Earlier I discussed the 'inside-out' operation, which is in fact the Fourier Transform. Since the times and frequencies are interchanged, species 4-7 become species 8-11 and vice versa. The effect of this is that multiplication by a species-4 atom is equivalent to convolution with a species-8 atom and so on. Species 0-3 and 12-15 stay the same species when Fourier Transformed.

Species 4 is a group of frequencies, such as a temperament, a chord, or a scale. Species 8 is a set of times, as in rhythm, repetition structure, or martellato. Thus, the Fourier Transform of a chord is a rhythm. Likewise, species 5 is a waveform shape, AM, tremolo, or filtering. – species 9 is a rhythm, a simple envelope, reverb, or resonance. Species 6 seems useless – species 10 is a type of rhythm. Species 7 is a symmetrical tone – species 11 is a rhythm, a control envelope, or a harmonic.

8.7 Implementation – PC

8.7.1 Synthesis

Synthesis based on these control structures was implemented in Turbo C on the PC. The creation of music from an aggregate of shorter acoustic events would generally fall into the category of *granular synthesis*. The approach outlined here differs from conventional granular synthesis in that quanta do not have a start or end, are (conceptually) infinitely long, and only contain a single frequency.

As memory is crucial in the context of music preparation, the previously written virtual memory drivers were used for the sample buffer. On 'Dan', my PC, this allowed the use of 8 MB of extended memory followed by disk space. The memory requirement naturally also depends on the sample rate, and any sample rate can be supported. A 6-kHz sample rate may not be high fidelity, but it shortens testing time significantly. Far memory is reserved for storage space for the parameters of the quanta.

Quanta are converted to samples using the recursive method described earlier, which needs 4 real multiplies and one real add per sample. This leads to fast synthesis, even on standard PC hardware.

The wavelet need only be calculated to the desired precision. In the earliest implementation, the only output device supported was the PC speaker, so 8-bit `char` accuracy is excessively high-fidelity! 16-bit

integer-coding is both commonplace and readily supported, so it is most convenient to operate with 16-bit integers. 24 bits may be preferable but are less easily supported – 32-bit words are supported as doubles, but this is of course costlier in memory.

We synthesise each sample from when it becomes significant (i.e. greater than 2^{-16}) to when it becomes insignificant again. This means that for a given alfa, a quantum that has a lower magnitude is calculated over fewer samples than one at a larger magnitude. This allows savings in execution time since many quanta will indeed have magnitudes well below the maximum. The relative freedom from resolution also permits the user to operate in 'draft mode' (8-bit or 12-bit) to allow faster working on a piece, but render the final output in 16 bits. It is also convenient to operate at a lower sample rate while working, as processing time is directly proportional to sample rate.

In the current implementation, each atom is broken into quanta, which are then calculated individually. This could be made more efficient by implementing synthesis routines designed for each species. For example, a species-4 atom could be made using N frequency calculations but only one Gaussian calculation per sample.

Since the start time and end time can be calculated from the time, density, and magnitude, the scheduling of synthesis is relatively straightforward.

8.7.2 C-based composition interface

The first available composition interface is to use the same language as the synthesis engine, in this case C. This means we can script a composition using the full range of C control structures and algorithms. However, this method also has the drawbacks that it is less intuitive and that compilation causes a sizeable delay between conceiving a musical concept and hearing it.

The central call is `MakeFillAtom`. This is illustrated best by example. The function call `MakeFillAtom(4,4,1,1, 0,1,2,3, 800,900,1000,800, Adur(1), 0.3)` means 'make an atom with 4 times, 4 frequencies, one alfa, and one magnitude, and fill it with the following values', and is equivalent to Q({0,1,2,3},{800,900,1000,800},Adur(1),0.3). This example gives the first four notes of 'Frère Jacques', and `Adur(dt)` is shorthand for π/dt^2. It is redundant but simpler to use the '4,4,1,1' notation than 'species 12 of size 4', although it is the latter form that is used internally.

Only a few simple functions need be implemented – multiplication and addition of atoms, as discussed earlier, and 'direct' multiplication and addition to their four arrays. For example, `Dmult(atom, 1, 1, 1, 4)` means 'multiply the times, frequencies, alfas, and magnitudes by 1,1,1,and 4 respectively' – i.e. 'amplify by 4'.

Higher-level operations are generally straightforward. The following pseudo-code illustrates note-level composition, and is relatively close to the actual syntax.

```
FrereTune=...
FrereTemp=Add(Frere,Dormez,Sonnez,Ding)
FrereJacques=Add(FrereTemp,DelayedCopy(FrereTemp,4))
Round=Convolve(FrereJacques, FourTimes)
Sound=MyFunction(Round)
Out=Convolve(Sound, RoomResponse)
Stretch(Out,44100)
Amplify(Out, 32767)
Synthesise(Out)
```

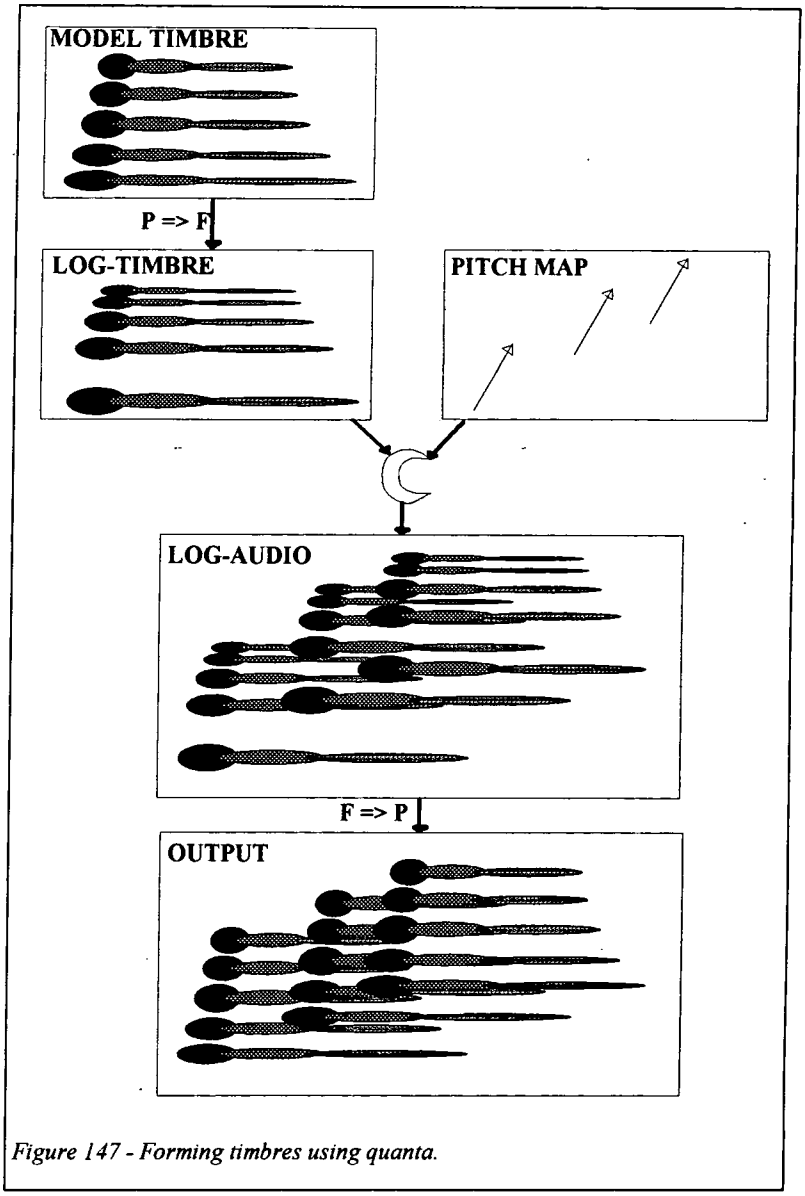


Figure 147 - Forming timbres using quanta.

With the structural expressiveness afforded by a general-purpose computing language, and the wide range of musical structures that can be represented simply as atoms, one can easily imagine many applications to composition.

It is necessary to carry out these operations in both the frequency and the log-frequency domain; this allows us to apply timbre, in a manner similar to that suggested by Mont-Reynaud and Tanguiane.[Mont-Reynaud 90, Tanguiane 95] The procedure for applying timbres is illustrated in Figure 147. The timbre, in the frequency domain, is converted into the pitch domain using the operation 'P→F'. The pitches are probably originally specified in the pitch domain. These are used to offset the timbres, giving the 'log-audio'. Then the 'F→P' operator converts these into frequencies.

The construction of larger musical structures is illustrated below. I show the musical intention and the atom that realises it. The figure in brackets is the species of the atom.

We could take a 12-bar bass line,	Q({0,1,2,3,4,5,6,7,8,9,10,11},pitch{0,5,0,0,5,5,0,0,7,5,0,0},∞,1)	*(12)*
convolve it with major chords,	Q(0,pitch{0,4,7},∞,1)	*(4)*
that have short upbeat notes,	Q({-0.125,0},0,∞,{0.1,1})	*(9)*
add a bass figure,	Q({0,0.5},pitch{-24,-12},∞,{1,0.8})	*(13)*
and put the lot into G,	Q(0,196,0,1)	*(0)*
emphasising phrases,	Q(0,{0,0.25},0,{0.8,0.2})	*(5)*
then form a guitar envelope,	Q(0,0,{10000,1000,100},{0.1,0.5,2})	*(3)*
make a timbre from an odd component,	Q(0,{1,3,5,7,9,11,13},0,{20,20,15,10,3,1,1})	*(5)*
and an even component,	Q(0,{2,4,6,8,10,12},0,{7,12,6,3,4,2})	*(5)*
scaling the amplitudes sensibly,	Q(0,0,1,0.01)	*(0)*
then applying a couple of formants,	Q(0,{2000,5000},1000000,{6,2})	*(5)*
and a little tremolo,	Q(0,{0,6},0,{0.95,0.05})	*(5)*
before adding reverberation,	Q({0,0.1,0.3},0,∞,{1,.2,.1})	*(9)*
and playing it to the left channel,	Q(0,0,1000000,0.9)	*(0)*
and the right channel one ITD later.	Q(0.0007,0,500000,0.4)	*(0)*

The entire example could be described in around 480 bytes as follows.

```
pitch(Q({0,1,2,3,4,5,6,7,8,9,10,11},pitch{0,5,0,0,5,5,0,0,7,5,0,0},∞,
1)⊗((Q(0,pitch{0,4,7},∞,1)⊗Q({0.125,0},0,∞,{0.1,1}))+Q({0,0.5},pitch
{-24,-12},∞,{1,0.8}))⊗Q(0,196,0,1))⊗Q(0,{0,0.25},0,{0.8,0.2})⊗(Q(0,0
,{10000,1000,100},{0.1,0.5,2})⊗Pitch(Q(0,{1,3,5,7,9,11,13},0,{20,20,
15,10,3,1,1})+Q(0,{2,4,6,8,10,12},0,{7,12,6,3,4,2})*Q(0,0,0,0.01))⊗Q
(0,{2000,5000},1000000,{6,2})⊗Q(0,{0,6},0,{0.95,0.05})⊗Q({0,0.1,0.3}
,0,∞,{1,.2,.1})⊗pan(Q({0,0.0007},.0,{100000000,50000000},{0.9,0.4}))
```

As an alternative to C, it may be worth examining the possibilities of implementing a formal grammar to expand expressions such as the one above.

8.7.3 Graphical User Interface

An attractive alternative to typed input is to use graphical output and mouse input. To this end, a graphical front-end with menuing and mouse support was designed. Figure 148 is a screen shot. The mouse is used both to operate the menuing system and to 'draw' quanta on the screen. For each quantum, four parameters must be specified (ignoring the imaginary part of magnitude). The time and frequency depend on the (x,y) position when the mouse is clicked, and the density and magnitude depend on where it is released.

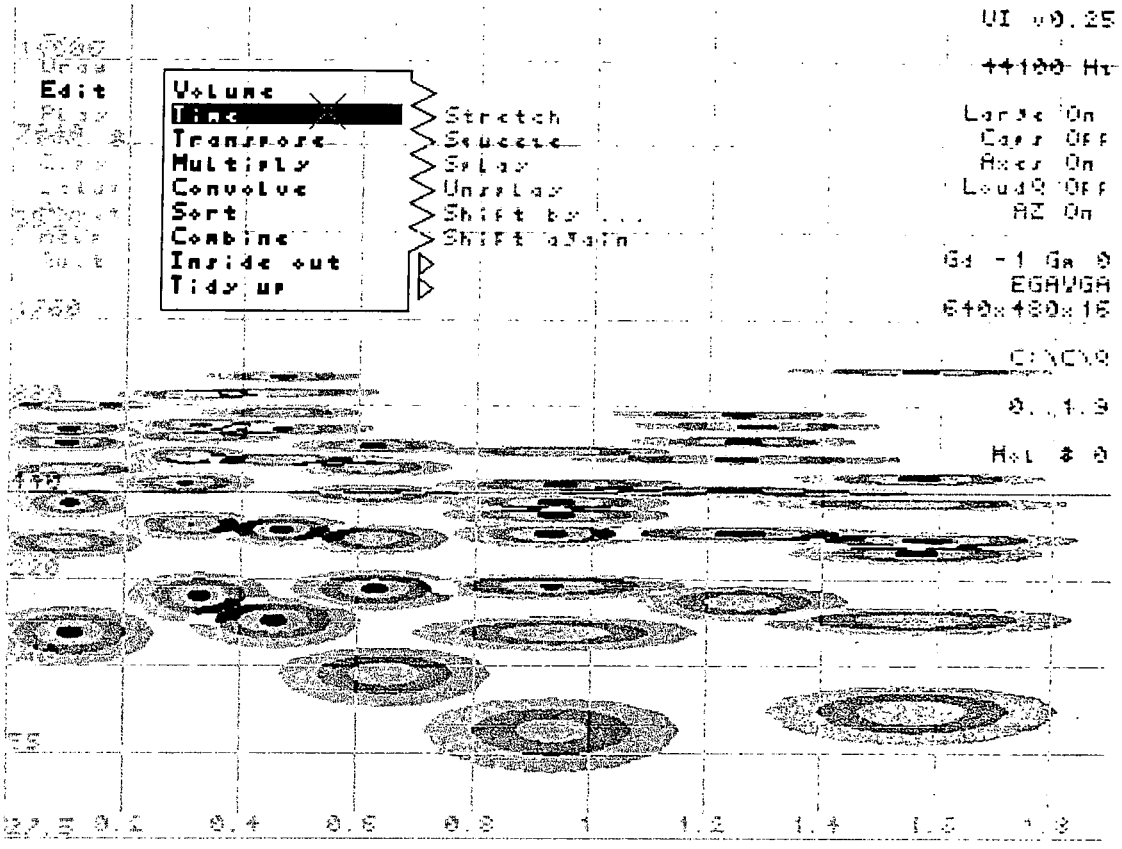


Figure 148 - Screen shot from the User Interface.

The figure above also illustrates the menu system. The menu edge points out when there is a submenu, and if the cursor is over such an item, the submenu text is displayed. When the menu item is a command, there is a separated triangle, as with 'Inside Out' and 'Tidy Up'. If the item has no action, not shown above, the right-hand edge is vertical. Characters are draw using a 5×4 font, the minimum for legibility.

The time resolution that can be drawn depends ultimately on the mouse resolution but also depends on the graphics resolution. The standard VGA resolution of 640*480 is adequate but not luxurious, so provision was made for supporting SVGA graphics drivers by using add-in BGI (Borland Graphics Interface) files. This allowed software support for resolutions up to 1280×1024, although the monitor specifications limited this to 1024×768.

As mentioned earlier, the parameters for the quanta are stored as 32-bit floating-point numbers. This means that a species 15 atom will take up 5×32/8 = 20 bytes of memory per quantum, plus a few bytes for the atom itself. The maximum number that could be loaded was around 7700, using 154000 bytes of far memory.

The system allows (arbitrarily) 21 molecules (only of species 15) to be manipulated; both linear operations (e.g. multiplication and convolution) and non-linear operations (e.g. time-stretching and transposition) are selected from the menu.

The graphical approach provides an intuitive way to compose sounds but lacks the structural generality of a procedural language at the higher levels required for composition. The ideal may lie between these two forms, incorporating links between entities such as in the Max[Puckette 90, Puckette 91b] or Nyquist[Dannenberg 93b] interfaces, or those illustrated by Desain's Domino system.[Desain 93] The underlying structures are identical, but we may wish to specify them by graphical drawing *or* by menu selection *or* by text input.

8.8 Implementation – C40

Synthesis of quanta, but not their control structures, has also been implemented on the C40 system with the DAC board developed by Milos Kolar, in a preliminary investigation of the feasibility of a real-time system. In principle, four multiplies and one addition need be performed for every output sample. We have previously shown that the maximum possible operations is 567 per sample (at 44100 Hz), but the C40 can carry out some operations in parallel. This suggests that we can calculate several hundred overlapping quanta in real time, although other overheads will reduce this.

A simple program was written to test the timing. The results given in Figure 149 show the running time as a proportion of real time for a given number of quanta, for a sample rate of 44100 Hz. It shows that we can calculate 220 quanta in real time if they are then ignored, but to calculate and send the samples individually to the DAC, we can only manage 87 in real time. Thus, if buffering is implemented we can expect a figure between 87 and 220. While this figure is not extravagant, this is sufficient for useful experiments with real-time synthesis.

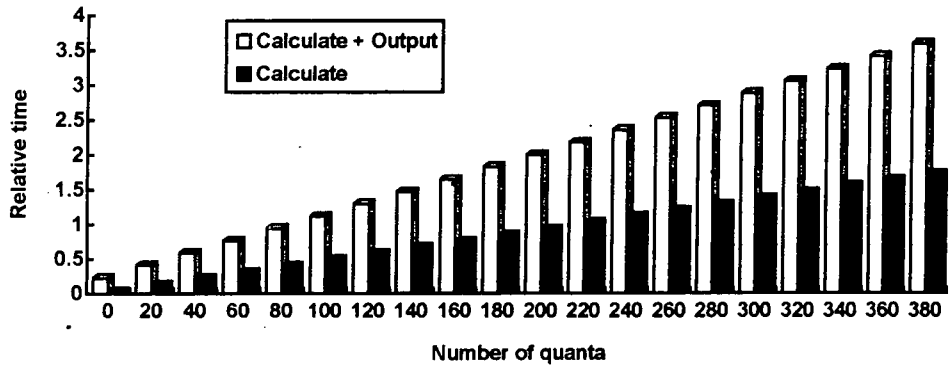


Figure 149 - Relative time for calculating N quanta on the C40.

8.9 Analysis using quanta

In discussing quanta above, I have concentrated on their applications to synthesis. However, they are also well suited to analysis, where we wish to form quanta from a given waveform. There is an infinite range of sets of quanta that sum to the input signal, and several methods of picking sub-optimal sets.

Below I outline several methods of analysis using quanta. Preliminary investigations have involved the implementation of both the Gabor transform and DC analysis. Further work is necessary to evaluate the effectiveness of these analysis schemes.

8.9.1 Gabor transform

The Gabor transform is essentially the same as the STFT with a Gaussian windowing function. This lets us represent a signal as quanta with times corresponding to the centre of each block and frequencies corresponding to the STFT bins. Another possibility is to combine this approach with the multirate techniques used in the previous system.

8.9.2 Sinc modelling

In this method, each sample is treated individually. Given a single sample and the knowledge of the sampling rate, we wish to convert it to the continuous form. This means convolving the input with a set of quanta that sum to sinc(x). Since $\text{sinc}(x)=\sin(\pi.x)/(\pi.x)$ and $\sin(\pi.x)$ is straightforward to express using quanta, we thus try to form $1/(\pi.x)$.

A good approximation to $1/(\pi.x)$ is given by summing

$$t_n = (\text{sign}).(\sqrt{2})^n \quad f_{all} = 0 \quad alfa_n = 2^{-n} \quad mag_n = (\text{sign}).(\sqrt{2})^{-n}.\kappa$$

from $n=-\infty$ to $+\infty$ and with *sign* taking values -1 and +1. Kappa is a constant, $\kappa \approx 0.07386$. Although this is a good approximation, it is not an equality. (The error is below the 14-bit level. In fact $\sqrt{2}$ was a lucky first guess and is not the best factor.)

When this is multiplied by $\sin(\pi.x)$, represented by two quanta, we get a sinc. For large positive n, these terms add distant broad small quanta, and the largest n worth using is the last with an amplitude greater than the precision desired – for 16-bit accuracy this would be around 32. For large negative n, they add narrow high quanta near x=0. There are inevitably problems at $x\approx0$, where 1/x swings from $-\infty$ to $+\infty$. In fact a truncated sum tends towards the sinc function except at x=0, where it remains stubbornly at zero since our approximation to 1/x is zero at x=0. (It is somewhat ironic that a sample at $t=t_0$ can be translated to its continuous representation at all times except t_0.) The resultant sinc approximation thus has a notch at x=0 whose width can be made arbitrarily small. To compensate for this notch, another term can be added at x=0.

The problem with such accurate modelling is that it takes a large number ($\approx$100) of quanta to model a sinc. A possible alternative route is by noting that $\text{sinc}(x)=\cos(x)\cos(\frac{1}{2}x)\cos(\frac{1}{4}x)\ldots$.

8.9.3 Iterative refinement

In the two methods above, we get a description in terms of a large number of wavelets with parameters chosen from a *finite and discrete* set. However, while they sum to the original signal, they do not necessarily correspond to the most compact description. If the Gabor transform is used initially, we

essentially inherit the same problems as the STFT and multirate STFT. One of these is that the time resolution is unable to capitalise on long periods of steady-state or very slow activity. The converse of this problem is that the time resolution may not be small enough to capture short details in the signal.

One possible solution to both these problems is to try to get this set to evolve into a better (i.e. smaller) set. I have described above how a quantum can be broken into several shorter quanta, and conversely how we can combine quanta into longer entities. Here the perfect symmetry in time and frequency of the Gabor wavelet is clearly better than time-limited and band-limited wavelets.

The STFT offers a static non-constant Q. The multirate system allows a static near-constant Q. The Multiresolution FT gives a discrete set of Q's to choose from. The Gabor transform combined with iterative refinement can allow a dynamic arbitrary Q, suited to exactly what is in the waveform. Likewise, the STFT, multirate STFT, and MFT offer a fixed set of times and frequencies, but the method I have described offers an arbitrary range of parameter values.

The optimisation could be implemented as an iterative system wherein a large pool of quanta interact by breaking up and combining with others. This suggests an 'artificial life' type genetic algorithm implementation in which the over-riding principle is to minimise the population. This has, however, not been implemented yet.

8.9.4 Basis pursuit

Another approach is to look for a more optimal set to begin with. Scott Chen outlines a technique known as Basis Pursuit.[ChenS 94, ChenS 95, ChenS 96] Chen stresses that each representation is non-unique, adding "It gives us the possibility of adaptation, i.e. of choosing among many representations one which is most suited to our purposes." He builds on Mallat's approach called 'Matching Pursuit', which iteratively removes elements from a signal.[Mallat 93] Gribonval's work based on this allows the decomposition of piano notes into a highly efficient representation.[Gribonval]

8.9.5 DC analysis

A similar approach involves solely time-domain analysis to begin with. It is clear that as the waveform is being read in, it would be relatively straightforward to iteratively look for and remove 'DC' Gaussians, i.e. quanta with f=0. We would then use the iterative principles discussed to evolve towards the frequencies. To illustrate, imagine the waveforms of the trombone notes shown earlier being input to an analysis system. Rather than carry out spectral analysis, we remove peaks from the waveform. The roughly periodic nature of the tone means that we would spot roughly the same Gaussian at roughly equally spaced times, with magnitudes roughly according to another Gaussian. We then replace each with a Gaussian with more accurate parameters (plus several error Gaussians) and convert this set into a set of harmonically related quanta over a longer period. Each set of Gaussians extracted similarly forms sets of quanta. We would then have several quanta over each partial, and these would then be combined into fewer quanta at the same frequency. The small errors arising from these operations would be put

back into the general 'pool' of things to be considered and recombined – they may cancel out errors from other operations.

8.10 Advantages for analysis and resynthesis

Many parts of the previous transcription system developed with the multirate STFT could be reused for quanta.

The second approach has a key advantage over the first, in that envelopes can be represented more compactly than a block-based approach, but without the inaccuracies introduce by straight-line approximations. This points to good potential for compression, although this has not been the main focus. Partial tracking could be much more efficient than with the multirate STFT using the iterative process described above. In a sense, the grouping principle of temporal continuity can be implemented by the model trying to reduce the size of a data set. Recall the continuity illusion, in which a quiet sinusoid was assumed to be present during loud noise, even if it is not. The smallest representation would be one in which the sine was coded as continuous. Even if it is not actually present, this would still apply.

Another problem in the original analysis has been solved. The eventual entities for a single partial were often combinations of different overlapping sets of blocks at different sets of frequencies, and it was not possible to merge these into a single entity. However, with quanta it is possible to change a quantum at t=5.997 seconds and f=882 Hz into one at t=6 and f=880 if higher-level knowledge (such as from other partials) suggests that this is more accurate, since we can derive the error precisely in terms of other quanta.

The distinction between bottom-up and top-down processing has been discussed before. Here the iteration allows both bottom-up and top-down processing of data. Yet whereas many top-down approaches go outside the top of the computer, and require a user to help the process, the principle of data minimisation permits an automatic system.

It is hoped that the results of such a system would be at least as good as those for the previous system. Indeed, the facts that the representation is more well behaved and that transformation is compactly encoded suggest that Gabor wavelets would outperform the earlier system.

The final benefit is that the synthesis is more efficient than before. With the block-based system, resynthesis would not have been practicable. As we have interpolated the times and frequencies, the advantages of the multirate STFT are no longer available.

8.11 Summary

Gabor wavelets, or quanta, offer much potential for synthesis. As well as being conceptually simple and computationally inexpensive, they lend themselves to an attractive interpretation as elemental sonic entities.

Another key advantage from the synthesists' viewpoint is that higher-level entities, such as melodies, timbres, scales, envelopes, filters, and reverberation, can also be expressed using the same paradigm. This contrasts with many synthesis methods where the 'score' and 'orchestra' are specified in completely different ways. The convenience of the quanta-based approach for specifying musical mid-level and high-level entities has been demonstrated.

However, the greatest promise in a quanta-based approach is that it appears to be as well suited to analysis as synthesis. Quanta can be used throughout the analysis scheme; they are applicable to both short and long spectral details. Although the complete analysis/resynthesis system has not yet been fully implemented, it seems that Gabor wavelets are well suited to this task.

9. Conclusions

In this research I have examined the analysis and resynthesis of polyphonic music, with the aim of developing a computer system for the characterisation and transcription of both notes and timbres. I have designed, implemented, and tested such a system.

The first stage of this research was an examination of the human auditory system and our perception of sound in general. This was followed by a discussion of our perception of musical sound. As some type of source separation is an essential part of polyphonic analysis, special attention was paid to our ability to interpret a complex waveform as the sum of its parts. The complex relationships between the parameters by which we characterise individual notes and the measurable physical parameters were explored.

This was followed by an investigation of various analysis and synthesis schemes, and the definition of the computational task to be addressed. It was noted that many audio processing applications are designed entirely for synthesis and have no corresponding analysis scheme, while others are designed entirely for analysis but do not permit resynthesis. The system under development is not intended as an analysis engine or as a synthesis engine but as a tool for analysis, transformation, and *re*synthesis. It has been designed to be applicable to complex musical situations without any prior knowledge of the input. Furthermore, it should be able to accurately resynthesise the original data.

The primary objective is the recognition and transcription of musical audio. This has applications in intelligent tools for auditory scene analysis and electroacoustic composition. An artificial listener can also be used in interactive performance applications such as auto-accompaniment, and in expressive performance analysis where the interpretation of a score is to be studied. Other related applications include source separation, noise removal, and data compression. Furthermore, while the system is not intended to be a physiologically accurate model of our auditory system, it does permit the emulation of human perception of music.

Both analysis and synthesis can be viewed as the conversion of one representation into another. Consequently, the characteristics of various existing representations of musical audio were compared. The most desirable attributes of a representation are that it is general, compact, parallel, and intuitive. Using these guidelines, many common synthesis techniques were found to be unsuitable for synthesis-by-analysis. Two possible approaches to the goal of analysis and resynthesis emerged; one is based on a multirate implementation of classical additive sine-wave synthesis, and the other is based on the newer field of wavelets.

Next I outlined the computational requirements of the task. This was followed by an examination of the capabilities of the computing platforms available; the IBM PC, the Texas Instruments' TMS320C40 digital signal processor, a network of Inmos transputers, and a Unix machine. The high computational

demands suggested that real-time operation would be unfeasible on any of these platforms. Accordingly, the overall design strategy was to adopt a modular pipelined configuration with a view to possible future implementation in real time on more powerful hardware. Each platform offered different benefits and drawbacks, and the route adopted was to develop a hybrid system. This used the C40 for the most computationally intensive stage and the PC for other stages where either graphical display or sound output was of greater importance.

This was followed by a detailed review of previous research in transcription and source separation. Distinctions between lines of research were made on the basis of the source polyphony, on whether they were designed for real-time applications, and most importantly on how much score and timbral information is known to the system in advance. Many of these had achieved some degree of success at transcribing a limited range of examples. It was noted that the lack of a standard set of test pieces makes it difficult to quantify the performance of a particular system, and harder still to compare the performance of different systems.

After this the various parts of the transcription system implemented were described in detail. For the initial stage of the analysis, a multirate Short-Time Fourier Transform was adopted. This permits a much more equitable allocation of the time-frequency bandwidth than the standard STFT. This stage of processing is carried out on the C40. This is followed by several further stages of processing on a standalone PC. First, prominent frequencies are extracted from the multirate STFTs. Next, these are tracked in time to form partials. The final process simulates the fusion of sets of harmonically related partials into notes.

Throughout the analysis, it was found to be extremely useful to be able to graphically display the data in its various representations. Techniques were developed for generating animations depicting the evolution in time of the multirate spectrum or the dependence of the processing on particular parameters.

The whole analysis is controlled using a specially designed interpreter for script files. The potential obstacle of the limited memory of the PC was addressed by implementing a virtual memory driver that allocates space from conventional memory, extended memory and hard disks. Each stage of the analysis is run as a separate program, also for reasons of memory. The undesirable side-effect of this design means that the stages cannot interact and hence the data flow is entirely bottom-up, so knowledge gained at later stages cannot be applied to earlier measurements.

The completed analysis system was then subjected to extensive testing. In order to identify the limitations of the system, a wide variety of sources was chosen. There were up to four methods of assessing its performance. The first was to listen to the MIDI resynthesis. In cases where the correct score was available, it was possible to compare scores in common practice notation. Where the correct

timing information was also available, it was possible to compare 'piano roll' scores. The last of these methods permitted quantification of the accuracy based on the proportion of overlap between two such piano rolls.

The system achieved a reasonable degree of success at transcribing polyphonic music. In particular, its success at unscrambling nine-note organ polyphony was greatly encouraging. However, application to other examples, including a brass trio, polyphonic piano, orchestral strings, solo didgeridoo, and cathedral bells, revealed that the system was not as robust or as general as had been hoped. The presence of noise, weak fundamentals, inharmonic tones, rapidly varying envelopes, and short low notes raised new complications.

These issues were addressed by redesigning the analysis system using a different fundamental building block, the Gabor wavelet. This is much more mathematically well behaved, in the sense that the wavelet and all its derivatives are continuous. In addition, it possesses the minimum time-bandwidth product permitted by the uncertainty relation. These wavelets can be synthesised through a very efficient recursive process. A technique was devised that allowed the compact storage and manipulation of multiple wavelets.

It was established at the outset that this scheme is practical for synthesis. This was demonstrated by the implementation of a text-based acoustic compiler and a graphical composition interface. Both of these programs highlighted an important advantage for composers of electroacoustic music – higher-level musical structures such as melodies, chords, and rhythms can be specified compactly using the same paradigm as that used to form timbres. In addition, transformative operations, such as filtering, reverberation, and timescale modification, can be carried out in the symbolic domain rather than on the waveform itself.

The various stages of the wavelet analysis have been examined, and seem to be practicable. Gabor wavelets can be merged and adjusted in a way not possible with the previous block-based approach, and this is expected to greatly improve the performance of partial tracking. Whereas the first system used an essentially static Q, wavelets permit Q to be changed dynamically to allow the efficient coding of both long and short musical entities. The remaining stages of analysis, such as harmonic matching, are equally feasible with a quanta-based system. Finally, the resynthesis stage is more efficient than with the previous system.

It is anticipated that these modifications will significantly improve the characterisation performance. However, further research is required to fully implement and evaluate this method of analysis, transformation, and resynthesis.

The main finding of this research is that it is possible for a computer to emulate the perception of polyphonic music. A more effective analysis will require a combination of bottom-up and top-down

processing. For robustness, it is also important for the analysis to be able to automatically adapt itself to the nature of the data. Graphical output is of vital importance for interpreting the large amounts of intermediate data. The wavelet approach outlined seems to have several clear advantages over block-based methods.

Polyphonic transcription is widely recognised as being one of the most elusive goals in computer music research, and I would not claim to have yet reached this goal. The key contributions I have made through this work are a thorough examination of the task, the complete implementation of a system for this purpose, and the presentation of results from testing its performance on a wide range of musical examples. I have also identified many common musical situations that require special attention, and highlighted methods through which this work may be taken forward by enhancements to the original design.

10. Appendices

10.1 Appendix A – Smooth FFTs and twisted butterflies

The general flow diagram for the radix-2 decimation-in-time FFT[Elliott] is shown in Figure 150 for N=8. (The diagram for decimation-in-frequency is the mirror image of this.) Xa is the input sample, Xb and Xc are intermediate results, and Xs is the spectrum data.

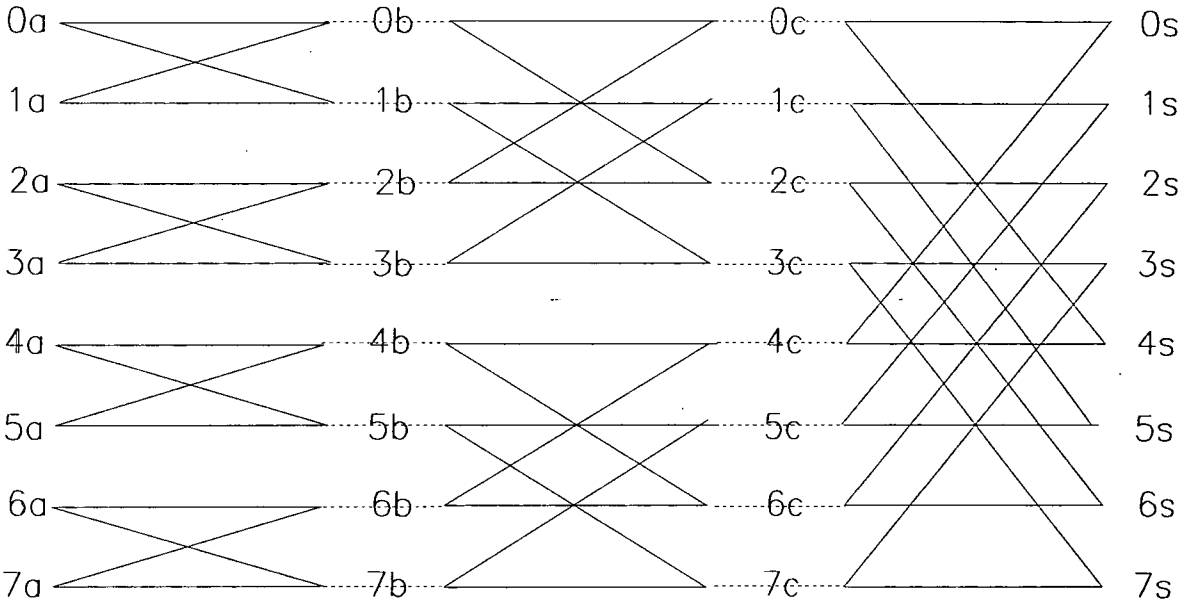


Figure 150 - Standard butterfly diagram.

The basic operation involves replacing x and y with x+y and x-y, and is known, from its shape, as a butterfly. The standard FFT requires $O(N.\log_2 N)$ calculations to be done at the end of the block. In order that this can be calculated with $O(\log_2 N)$ operations each sample, we must distort each of the butterflies as shown in Figure 151, which illustrates three consecutive FFTs. This rearrangement is similar to the pipeline FFT.[Rabiner 75]

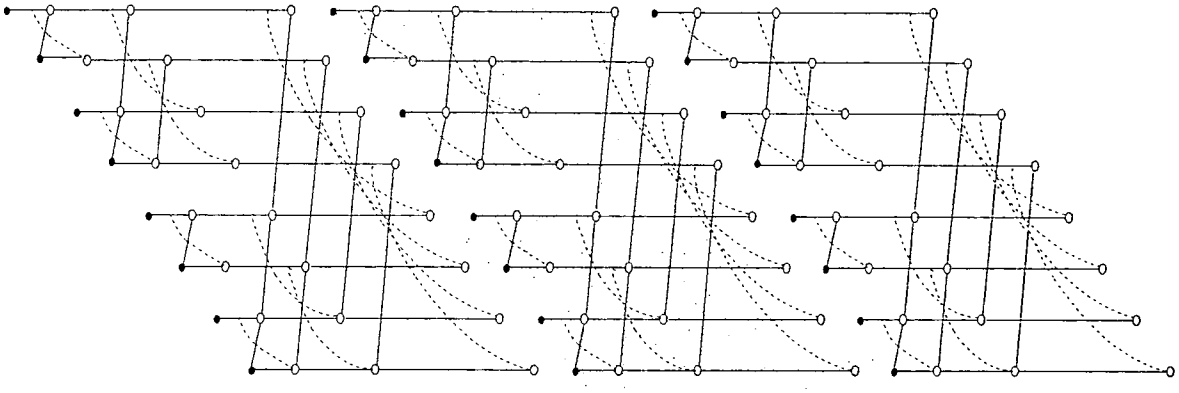


Figure 151 - Twisted butterfly diagram.

Note that the intermediate results must be stored, as shown by the dotted lines, because we require the value of '2a' after we have calculated '2b'. To see how much extra memory we require, note that at any time, there is one 'a' value in storage, two 'b' values, and four 'c' values. By extension, we can show that the total extra storage is equal to N-1.

Although the computation is relatively smooth with respect to time, the overhead in swapping values to and from these temporary locations fluctuates slightly. At stage s, the number of these memory accesses is given by the number of bits set in the binary representation of (s+1), up to a maximum of $\log_2 N$ (3 in this example).

The algorithm for N=8 is as follows:-

Stage	*Data input*	*Fetch*	*Calculate*	*Calculate*	*Spectrum output*	*Store*
	0a	6a	7b = 7a-6a	5c = 5b+7b	1s = 1c+5c	0a
	1a	4b	0b = 0a+1a	6c = 6b-4b	2s = 2s+6c	0b
	2a	0a, 5b	1b = 0a-1a	7c = 7b-5b	3s = 3c+7c	1b, 2a
	3a	0c	2b = 2a+3a	0c = 0b+2b	4s = 0c-4c	0c
	4a	2a, 1c	3b = 2a-3a	1c = 1b+3b	5s = 1c-5c	4a, 1c
	5a	0b, 2c	4b = 4a+5a	2c = 0b-2b	6s = 2c-6c	4b, 2c
	6a	4a, 1b, 3c	5b = 4a-5a	3c = 1b-3b	7s = 3c-7c	6a, 5b, 3c
	7a	-	6b = 6a+7a	4c = 4b+6b	0s = 0c+4c	-

Table 44 - Algorithm for smoothed FFT.

10.2 Appendix B - Derivation of DFT as filter

The DFT can be reinterpreted as a set of filters in two ways:-

Periodic DFT filter:-
A periodic frequency response of $\exp\left[-j\cdot\pi\cdot f\cdot\left(1-\frac{1}{N}\right)\right]\cdot\frac{\sin(\pi\cdot f)}{N\cdot\sin\left(\frac{\pi\cdot f}{N}\right)}$

convolved with a *nonperiodic* input spectrum.

Nonperiodic DFT filter:-
A nonperiodic frequency response of $\exp\left[-j\cdot\pi\cdot f\cdot\left(1-\frac{1}{N}\right)\right]\cdot\frac{\sin(\pi\cdot f)}{\pi\cdot f}$

convolved with a *periodic* input spectrum.

These are equivalent; the former is used below.

If the input is x(t), then we sample it by multiplication with N delta functions at a spacing of T. The DFT is thus given by

$$X(k)=\frac{1}{N}\cdot\int_0^P\sum_{(n=0..N-1)}\delta(t-n\cdot T)\cdot x(t)\cdot e^{-j\cdot 2\cdot\pi\cdot k\cdot\frac{t}{P}}\,dt$$

We can change the limits of integration to give

$$X(k)=\int_{-\infty}^{\infty}d(t)\cdot x(t)\cdot e^{-j\cdot 2\cdot\pi\cdot k\cdot\frac{t}{P}}\,dt \quad \text{where} \quad d(t)=\frac{1}{N}\cdot\sum_{(n=0..N-1)}\delta(t-n\cdot T)$$

Since the transform of a product is a convolution, $X(k)=F(x(t)\cdot d(t))=D\left(\frac{f}{P}\right)\times Xa\left(\frac{f}{P}\right)$

where Xa is the Fourier transform of the original analogue function. Thus,

$$X(k)=\int_{-\infty}^{\infty}D\left(\frac{f}{P}\right)\cdot Xa\left(\frac{k-f}{P}\right)d\frac{f}{P}$$

D(f/P) defines the filter response, and is the Fourier transform of d(t). Using P=NT, we get:-

$$D\left(\frac{f}{P}\right)=\frac{1}{N}\cdot\int_{-\infty}^{\infty}\sum_{(n=0..N-1)}\delta(t-n\cdot T)\cdot e^{-j\cdot 2\cdot\pi\cdot f\cdot\frac{t}{P}}\,dt=\frac{1}{N}\cdot\sum_{(n=0..N-1)}e^{-j\cdot 2\cdot\pi\cdot f\cdot\frac{n}{N}}$$

Using the formula for the sum of a geometric sequence, $D\left(\frac{f}{P}\right)=\frac{1}{N}\cdot\frac{1-e^{-2\cdot j\cdot\pi\cdot f}}{1-e^{-2\cdot j\cdot\pi\cdot\frac{f}{N}}}$

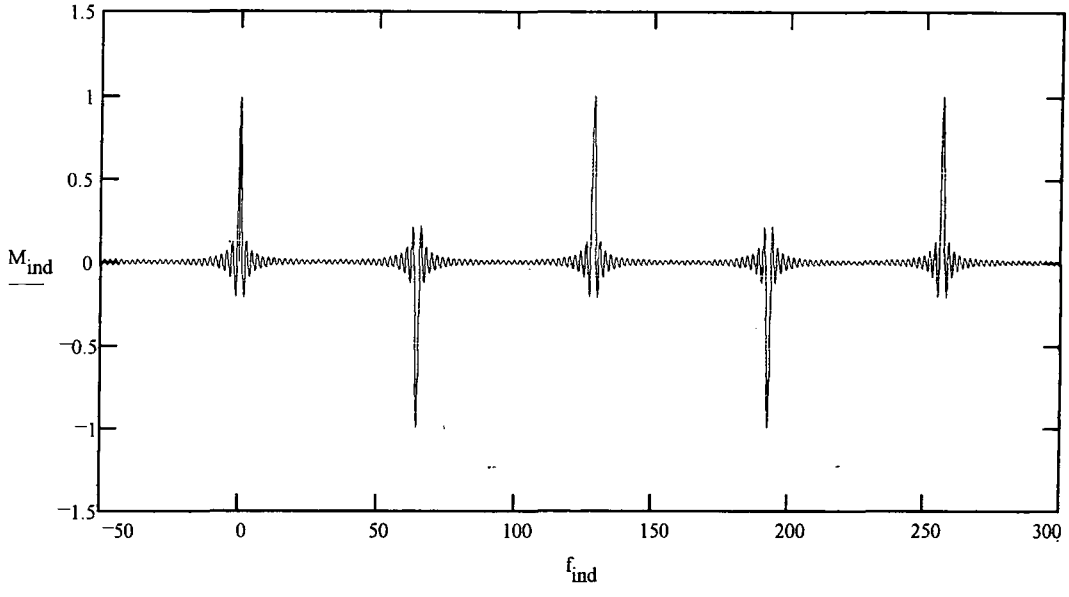
As this is independent of the period P, we will normalise it using P=1s. This gives the final result:-

$$D(f)=e^{-j\cdot\pi\cdot f\cdot\left(1-\frac{1}{N}\right)}\cdot\frac{\sin(\pi\cdot f)}{N\cdot\sin\left(\pi\cdot\frac{f}{N}\right)}$$

The first term here indicates the linear phase shift, and the second is the amplitude response, showing how each frequency leaks into neighbouring bins in the DFT.

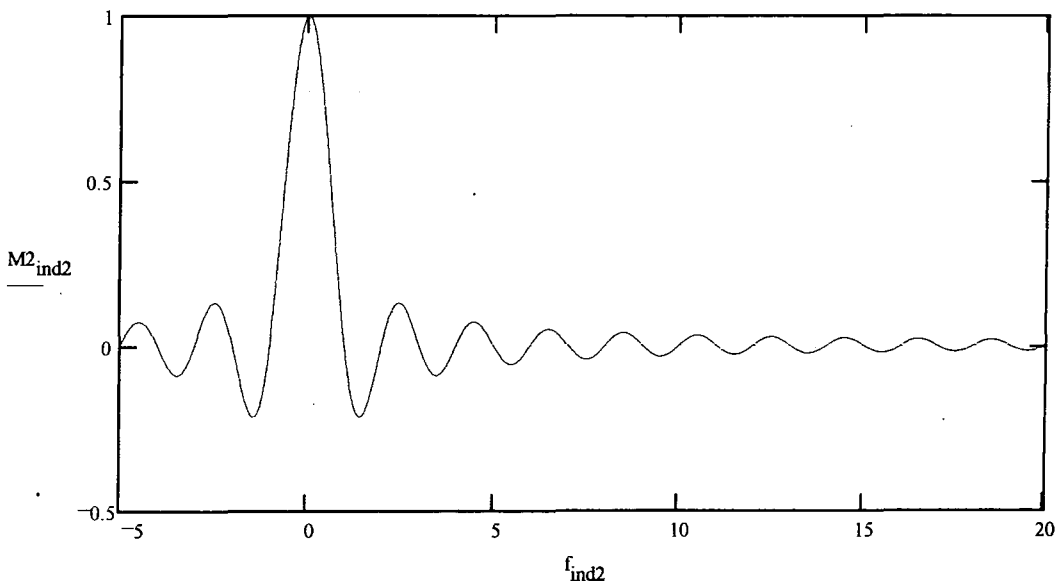
The graph below shows how the magnitude of D(f) varies with f:-

$N := 64 \qquad Z := 10 \qquad ind := 0..350 \cdot Z \qquad f_{ind} := -50 + \frac{ind}{Z} \qquad M_{ind} := \frac{\sin(\pi \cdot f_{ind})}{N \cdot \sin\left(\pi \cdot \frac{f_{ind}}{N}\right)}$



Note that the response shown above at f=128 is -1, but the phase shift turns this back to approximately +1. The first section of this is approximately equal to the sinc response of the nonperiodic filter, and is shown in more detail below:-

$N := 128 \quad Z := 50 \quad ind2 := 0..25 \cdot Z \quad f_{ind2} := -5 + \frac{ind2}{Z} \quad M2_{ind2} := \frac{\sin(\pi \cdot f_{ind2})}{N \cdot \sin\left(\pi \cdot \frac{f_{ind2}}{N}\right)} \quad M2_{5 \cdot Z} := 1$



10.3 Appendix C - Derivation of DFT deconvolution

In Appendix B, we showed that the DFT can be expressed as:-

$$X(k)=\int_{-\infty}^{\infty} d(t)\cdot x(t)\cdot e^{-j\cdot 2\cdot\pi\cdot k\cdot\frac{t}{P}}\,dt \qquad \text{where} \qquad d(t)=\frac{1}{N}\cdot\sum_{(n=0..N-1)}\delta(t-n\cdot T)$$

If we apply a weighting w(t) to the data (the term window is best kept for use in the spectral domain - i.e. a window is the transform of a weighting), then we must modify this to:-

$$X(k)=\int_{-\infty}^{\infty} d(t)\cdot w(t)\cdot x(t)\cdot e^{-j\cdot 2\cdot\pi\cdot k\cdot\frac{t}{P}}\,dt$$

$$X(k)=F(d(t)\cdot w(t)\cdot x(t))=D'\left(\frac{f}{P}\right)\times Xa\left(\frac{f}{P}\right)$$

where Xa is the Fourier transform of the original analogue function as before, and

$$D'\left(\frac{f}{P}\right)=F(d(t)\cdot w(t))$$

There are many possible weightings, derived for different purposes, making a compromise between numerical simplicity, mainlobe sharpness, maximum sidelobe level, and sidelobe fall-off. A large and useful category, including the Hamming and Hanning weightings, is defined by:-

$$w(t)=\sum_k a_k\cdot\cos\left(2\cdot\pi\cdot\frac{k\cdot t}{P}\right) \qquad \sum_k a_k=1$$

The weighting adopted was the 4-term Blackman-Harris weighting, which reduces sidelobes to -92 dB, at the expense of a relatively broad mainlobe. This is given by:-

$$a_0=0.35875,\ a_1=0.48829,\ a_2=0.14128,\ a_3=0.01168$$

The transform of the weighting function is thus:-

$$W(f)=a_0\cdot\delta(f)+\frac{1}{2}\cdot a_1\cdot\delta(f+1)+\frac{1}{2}\cdot a_1\cdot\delta(f-1)+\frac{1}{2}\cdot a_2\cdot\delta(f+2)+\frac{1}{2}\cdot a_2\cdot\delta(f-2)+\ldots$$

We can now derive D`(f) as $D'(f) = D(f) \times W(f) = \ldots$

$$\frac{a_0}{N}\cdot\frac{\sin(\pi\cdot f)}{\sin\left(\frac{\pi\cdot f}{N}\right)}\cdot\exp\left[-j\cdot\pi\cdot f\cdot\left(1-\frac{1}{N}\right)\right]\ldots$$
$$+\frac{a_1}{2\cdot N}\cdot\frac{\sin(\pi\cdot(f+1))}{\sin\left[\frac{\pi\cdot(f+1)}{N}\right]}\cdot\exp\left[-j\cdot\pi\cdot(f+1)\cdot\left(1-\frac{1}{N}\right)\right]+\frac{a_1}{2\cdot N}\cdot\frac{\sin(\pi\cdot(f-1))}{\sin\left[\frac{\pi\cdot(f-1)}{N}\right]}\cdot\exp\left[-j\cdot\pi\cdot(f-1)\cdot\left(1-\frac{1}{N}\right)\right]\ldots$$
$$+\frac{a_2}{2\cdot N}\cdot\frac{\sin(\pi\cdot(f+2))}{\sin\left[\frac{\pi\cdot(f+2)}{N}\right]}\cdot\exp\left[-j\cdot\pi\cdot(f+2)\cdot\left(1-\frac{1}{N}\right)\right]+\frac{a_2}{2\cdot N}\cdot\frac{\sin(\pi\cdot(f-2))}{\sin\left[\frac{\pi\cdot(f-2)}{N}\right]}\cdot\exp\left[-j\cdot\pi\cdot(f-2)\cdot\left(1-\frac{1}{N}\right)\right]\ldots$$
$$+\ldots$$

If we limit the series to four sinusoidal terms, and use the abbreviation $\operatorname{zinc}(f)=\frac{\sin(\pi\cdot f)}{\sin\left(\frac{\pi\cdot f}{N}\right)}$

then we finally have

$$D'(f)=\frac{\exp\left[-j\cdot\pi\cdot f\cdot\left(1-\frac{1}{N}\right)\right]}{N}\cdot\left[\begin{array}{l} a_0\cdot\operatorname{zinc}(f)\ \ldots \\ +\frac{a_1}{2}\cdot\exp\left(\frac{j\cdot\pi}{N}\right)\cdot\operatorname{zinc}(f+1)+\frac{a_1}{2}\cdot\exp\left(\frac{-j\cdot\pi}{N}\right)\cdot\operatorname{zinc}(f-1)\ \ldots \\ +\frac{a_2}{2}\cdot\exp\left(\frac{2\cdot j\cdot\pi}{N}\right)\cdot\operatorname{zinc}(f+2)+\frac{a_2}{2}\cdot\exp\left(\frac{-2\cdot j\cdot\pi}{N}\right)\cdot\operatorname{zinc}(f-2)\ \ldots \\ +\frac{a_3}{2}\cdot\exp\left(\frac{3\cdot j\cdot\pi}{N}\right)\cdot\operatorname{zinc}(f+3)+\frac{a_3}{2}\cdot\exp\left(\frac{-3\cdot j\cdot\pi}{N}\right)\cdot\operatorname{zinc}(f-3) \end{array}\right]$$

The numerator of the zinc function need only be calculated once, as the sin function repeats. Note also that care must be taken when the denominator is equal to zero; here, zinc(f) = +/- N.

(Some analyses ignore the exponential terms in this sum, giving

$$\overline{D'(f)}=\frac{\exp\left[-j\cdot\pi\cdot f\cdot\left(1-\frac{1}{N}\right)\right]}{N}\cdot\left(\begin{array}{l} a_0\cdot\operatorname{zinc}(f)+\frac{a_1}{2}\cdot\operatorname{zinc}(f+1)+\frac{a_1}{2}\cdot\operatorname{zinc}(f-1)\ \ldots \\ +\frac{a_2}{2}\cdot\operatorname{zinc}(f+2)+\frac{a_2}{2}\cdot\operatorname{zinc}(f-2)+\frac{a_3}{2}\cdot\operatorname{zinc}(f+3)+\frac{a_3}{2}\cdot\operatorname{zinc}(f-3) \end{array}\right)$$

However, this will lead to small but significant inaccuracies. There is little overhead in calculating this phase correction, as the later terms can be derived recursively.)

To deconvolve the spectrum, we use the above equation to evaluate the spectrum that would result from a sinusoid at a specified frequency, and calculate the correlation between it and the actual spectrum at seven neighbouring points. The procedure gives both a spectral estimate and a 'goodness of fit' coefficient. We then iterate until this fit is maximised.

10.4 Appendix D - Derivation of errors in resynthesis

Here we determine the time-domain errors caused by resynthesising data based on an estimate in the spectral domain.

If the original signal is given by $y=(A+a)\cdot\cos((\omega+\Delta\omega)\cdot t+(\Phi+\Delta\Phi))$

and the estimated signal is $y'=(A-a)\cdot\cos((\omega-\Delta\omega)\cdot t+(\Phi-\Delta\Phi))$

then the error is

$$e=y-y'=(A+a)\cdot\cos((\omega+\Delta\omega)\cdot t+(\Phi+\Delta\Phi))-(A-a)\cdot\cos((\omega-\Delta\omega)\cdot t+(\Phi-\Delta\Phi))$$

Note that, for numerical simplicity,

a	is *half* the amplitude error
$\Delta\omega$	is *half* the frequency error
$\Delta\Phi$	is *half* the phase error

We rearrange this as

$$e=(A+a)\cdot\cos((\omega\cdot t+\Phi)+(\Delta\omega\cdot t+\Delta\Phi))-(A-a)\cdot\cos((\omega\cdot t+\Phi)-(\Delta\omega\cdot t+\Delta\Phi))$$

For brevity, define $p=\omega\cdot t+\Phi$ and $\Delta p=\Delta\omega\cdot t+\Delta\Phi$

$$e=(A+a)\cdot\cos(p+\Delta p)-(A-a)\cdot\cos(p-\Delta p)$$

$$e=(A+a)\cdot(\cos(p)\cdot\cos(\Delta p)-\sin(p)\cdot\sin(\Delta p))-(A-a)\cdot(\cos(p)\cdot\cos(\Delta p)+\sin(p)\cdot\sin(\Delta p))$$

$$e=A\cdot\cos(p)\cdot\cos(\Delta p)-A\cdot\sin(p)\cdot\sin(\Delta p)+a\cdot\cos(p)\cdot\cos(\Delta p)-a\cdot\sin(p)\cdot\sin(\Delta p)\ \ldots$$
$$+\ -A\cdot\cos(p)\cdot\cos(\Delta p)-A\cdot\sin(p)\cdot\sin(\Delta p)+a\cdot\cos(p)\cdot\cos(\Delta p)+a\cdot\sin(p)\cdot\sin(\Delta p)$$

$$e=-2\cdot A\cdot\sin(p)\cdot\sin(\Delta p)+2\cdot a\cdot\cos(p)\cdot\cos(\Delta p)$$

$$e=-2\cdot A\cdot\sin(\omega\cdot t+\Phi)\cdot\sin(\Delta\omega\cdot t+\Delta\Phi)+2\cdot a\cdot\cos(\omega\cdot t+\Phi)\cdot\cos(\Delta\omega\cdot t+\Delta\Phi)$$

This is the main result; the first term corresponds to the errors in frequency and phase, and the second corresponds to the error in amplitude. Note that the terms are out of phase; one leads the other by a quarter of a cycle. Thus, the total error is less than the sum of the two errors. If we expand the first term again, we get the following expression for e:-

$$\underbrace{-2\cdot A\cdot\sin(\Delta\omega\cdot t)\cdot\cos(\Delta\Phi)\cdot\sin(\omega t+\Phi)}_{\text{frequency error}}\ \underbrace{-2\cdot A\cdot\cos(\Delta\omega\cdot t)\cdot\sin(\Delta\Phi)\cdot\sin(\omega t+\Phi)}_{\text{phase error}}+\underbrace{2\cdot a\cdot\cos(\Delta\omega\cdot t+\Delta\Phi)\cdot\cos(\omega t+\Phi)}_{\text{amplitude error}}$$

If the errors are small, then we can use the approximations $\sin(\varepsilon)=\varepsilon$ and $\cos(\varepsilon)=1$ to give

$$e'=-2\cdot A\cdot\Delta\omega\cdot t\cdot\sin(\omega t+\Phi)-2\cdot A\cdot\Delta\Phi\cdot\sin(\omega t+\Phi)+2\cdot a\cdot\cos(\omega t+\Phi)$$

10.5 Appendix E – Syntax of MEX script language

Below I formally define the syntax used by the script language interpreter MEXEC v0.27. Brackets [ ] indicate optional terms. Terms in **BOLD** are literals. The '!' character is signified by '!!'.

program	=	*commandlines*
commandlines	=	*commandline*
	=	*commandline commandlines*
commandline	=	*[linenumber] [command] [; [comment]]*
command	=	*assignment*
	=	*executestatement*
	=	*dosstatement*
	=	*ifstatement*
	=	*gotostatement*
	=	*echostatement*
	=	*pausestatement*
	=	*beepstatement*
	=	*endstatement*
	=	*commentstatement*
	=	*endcommentstatement*
assignment	=	*variablename* **=** *stringexpr*
stringexpr	=	*string*
	=	**!** *variablename*
	=	*string stringexpr*
	=	*stringexpr* **! &** *stringexpr*
executestatement	=	**EXECUTE** *stringexpr*
dosstatement	=	**DOS** *stringexpr*
ifstatement	=	**IF** *stringexpr* **=** *stringexpr* **GOTO** *linenumber*
gotostatement	=	**GOTO** *linenumber*
echostatement	=	**ECHO** *stringexpr*
pausestatement	=	**PAUSE**
beepstatement	=	**BEEP**
endstatement	=	**END**
commentstatement	=	**COMMENT**
endcommentstatement	=	**ENDCOMMENT**

10.6 Appendix F – Listing of transcription script file

Below is the script file ALL.MEX, used to control the transcription process.

```
; All.Mex - Music EXecutable
; version 9, 19 June 1995
;
;DOS mode con lines=50
ECHO All.Mex
; douglas nunn - started 23/6/94
;
; v8 - use megasort.exe
; 9 - add gpc, change default mex dir


;ECHO one two three    !& four !&     five   !&   six  !&       seven
; now implemented with mexec v0.27
; !& concatenates and removes whitespace


;-- host hardware etc ----------------------
;Machine=Wendy
Machine=Dan
;Machine=TLab

C40HostMachine=TLab

;------------

IF !Machine=Dan GOTO 20
IF !Machine=Wendy GOTO 25
IF !Machine=TLab GOTO 27
ECHO Error - what machine are you on?
END
20 ECHO setup for DAN
CDir=c:\c\          ; C directory   NB use trailing slash
WDir=D:\mex\        ; MEX dir
ParCDir=c:\tic1v1\
CakewalkDir=c:\cakewalk\
24 GOTO 29

25 ECHO setup for WENDY
CDir=i:\c\
WDir=m:\mex\
ParCDir=c:\tic1v1\
CakewalkDir=i:\cakewalk\
26 GOTO 29

27 ECHO setup for TLAB
CDir=i:\c\
WDir=m:\mex\
ParCDir=c:\tic1v1\
CakewalkDir=i:\cakewalk\
GOTO 29

29 ;

;-------------------------------------------

DefaultTask = minimendel
Task        = !DefaultTask

;Task=ppanther; FULL VERSION ON WENDY
;Task=minimendel ; re-do with fixed(?) pickout

Task=wmendel
Task=mendel

Task=poulenc
Task=aase

;Task=mtest1
;Task=mtest2

Task=bells
Task=gpc
```

```
Task=schum


doConvert    = no

doOsa        = no
;IF !Machine=C40HostMachine doOsa = yes

doDispSpec   = no;yes
doCharacter  = no;yes
doPickout    = no;yes
doA2B        = yes
doShowSlb    = yes
doTrack      = yes ; return error=junk? 21
doReorder    = yes; NOW WORKING - vmem018
doShowReo    = no;yes
doFT         = no;yes; ok for >13148, >68239 for minimend
doBattle     = no;yes ;
doMakeWork   = no;yes
doMakeMidi   = no;yes
doCakewalk   = no;yes

SampleRate      = 32000   ; DEFAULT


;--------

ECHO Setting up task ' !Task '

IF !Task=minimendel GOTO 45
IF !Task=mendel GOTO 50
IF !Task=ppanther GOTO 55
IF !Task=tooshort GOTO 60
IF !Task=poulenc GOTO 65
IF !Task=wmendel GOTO 67
IF !Task=aase GOTO 68
IF !Task=mtest1 GOTO 690
IF !Task=mtest2 GOTO 691
IF !Task=bells GOTO 692
IF !Task=gpc GOTO 693
IF !Task=schum GOTO 694

ECHO error - what task?
END

;------------
45 ECHO Setting up 'MINIMENDEL' task...

SampleRate=16000     ; for battle and others before (?)

MonoFile         = !WDir !& minimend\minimend.snd
doConvert        = no                          ; start with *.c40
;;;;;;;;;;;doOsa=no
SpectrumFile     = !WDir !& minimend\minimend.c40

49 GOTO 70

;------------

50 ECHO Setting up 'MENDEL' task...

MonoFile         = m:\mendel\mendel4.mon
doConvert        = no                          ; start with *.c40
;;;;;;;;;;;doOsa=no
SpectrumFile     = !WDir !& mendfull\spectra.c40

51 GOTO 70

;--------------
55 ECHO Setting up 'PPANTHER' task...

SampleRate=8000

SourceFile = m:\au\ppanther.au
Amplify= -v2.10
doConvert=no

SpectrumFile = !WDir !& ppanther\pp.c40
```

```
56 GOTO 70

;---------------
60 ECHO Setting up 'TOOSHORT' task...

SampleRate=44100

SourceFile=!WDir !& ts\tsmono.snd    ; ???
doConvert=no
doOsa=no
SpectrumFile = !WDir !& ts\tsmono.c40

GOTO 70

;---------------
65 ECHO Setting up 'POULENC' task...

SampleRate=16000

;;;;;;;;;SourceFile=!WDir !& ts\tsmono.snd    ; ???
doConvert=no
;doOsa=no
SpectrumFile = !WDir !& poulenc\poulenc.c40

GOTO 70

;---------------
67 ECHO Setting up 'WMENDEL' task...
; this one was done with a window

SampleRate=32000

;;;;;;;;;SourceFile=!WDir !& ts\tsmono.snd    ; ???
doConvert=no
doOsa=no
SpectrumFile = m:\spec\hamming.c40

GOTO 70
;--------------

68 SampleRate=16000
;;;;SourceFile=!WDir !& aase\aase.snd
doConvert=no
doOsa=no
SpectrumFile=!WDir !& aase\aase.c40
GOTO 70

;--------------
690   ; numbers don't have to be in order
SourceFile=!WDir !& mtest1\mtest1
SampleRate=16000
doOsa=no
;SpectrumFile=i:\mex\mtest1\mtest1.c40 ;wendy
SpectrumFile=!WDir !& mtest1\mtest1.c40

GOTO 70

;-------------
691   ; numbers don't have to be in order
SourceFile=!WDir !& mtest2\mtest2.snd
SampleRate=16000
doOsa=no
SpectrumFile=!WDir !& mtest2\mtest2.c40

GOTO 70

;-----
692
;;; SourceFile=!WDir !& bells\bells2.snd
SampleRate=16000
doOsa=no
SpectrumFile=!WDir !& bells\bells2.c40
GOTO 70

;-----
693
;;; SourceFile=!WDir !& gpc\gpc.snd
```

```
SampleRate=16000
doOsa=no
SpectrumFile=!WDir !& gpc\gpc.c40
GOTO 70

;-----
694
;;; SourceFile=!WDir !& gpc\gpc.snd
SampleRate=32000
doOsa=no
SpectrumFile=!WDir !& schum\schum.c40
GOTO 70


;-------------
70


; global default settings


; <<input sound file of any format>>

ConvertProg     = m:\music\sox\sox.exe
GenFlags        = -V
Amplify         = ; no amplification
SourceFlags     =   ; nothing needed - SOX can work it out
DestFlags       = -w  ; write as 16-bit words
MonoFile        = c:\waves\mex.wav


; for screen capturing, add the flag -m
UseMono = no
DispSpecFlags=  ; nothing
if !UseMono=yes DispSpecFlags= -m
; display spectra
DispSpecProg = !CDir !& readsp.exe !DispSpecFlags   ; new


; characterise

CharacterProg = !CDir !& distrib2.exe
CharacterFile = !WDir !& charactr.mex


; <<mono 16-bit *.snd file>>

C40LoaderProg  = !ParCDir !&tis.exe
C40Flags       =      ;
AnalysisProg   = i:\parallel\osa.app


; <<C40 spectra>>

PostProcMethod = old
PickoutProg    = !CDir !&pickout.exe
SineListA      = !WDir !& mex.sla


; <<Sine List - format A - unsorted>>

;;;;old;;;;;;;;;A2BProg        = A_to_B5.Bat

A2BProg         = MegaSort.Exe
A2BFlags        = -d
SineListB       = !WDir !& mex.slb


; <<Sine List - format B - sorted>>

; display list b
RunBasic = c:\dos\qbasic /h /run
ShowSlbProg = !RunBasic !CDir !&showslb3.bas
```

```
ChainFile       = !WDir !& mex.chn     ; link file
;;;TrackProg       = !CDir !&TrakSin5.Exe
TrackProg       = !CDir !&TrakSin6.Exe     ; all.mex v8


; <<Chain File AND Sine List B>>

;;;ReorderProg     = !CDir !&reorder.exe    ; old - thrashes disk
ReorderProg     = !CDir !&VReorder.Exe  ; NEW - uses virtual memory
; nb new one still buggy near end of file


ReorderedList   = !WDir !& mex.reo       ; reorder.52


; <<Reordered list of sines>>


; display reordered list
ShowReoProg = !RunBasic !CDir !&showtrx7.bas ;was 5 before



FTProg          = !CDir !&ft.exe
TempDumpName    = !WDir !& mex_dump      ; no extension


; <<chain, own, and seg files>>

BattleProg      = !CDir !&batl.exe
AsciiFile       = !WDir !& mex.asc


; <<ASCii format MIDI file>>

WorkFile        = !WDir !& mex.wrk
WorkEditFile    = !WDir !& mexedit.wrk
Asc2WrkProg     = !CDir !&makewrk6.bat     ; v6 now


; <<Cakewalk work file>>

MidiFile        = !WDir !& mex.mid
Cake2MidProg    = !CakewalkDir !& Cake2Mid.Exe


; <<standard MIDI file>>

Cakewalk        = !CakewalkDir !& CakePro.Exe
CakeFlags       = -50 ; 50-line screen



;  ##  #    # # # ### ##  ###
; #   # # ### # # ##  ##   #
;  ##  #  #    #  ### # #  #


; convert source file to the desired format
; we don't need monolyse.exe - sox can cope with more formats


IF !doConvert=no ECHO Skipping conversion to mono 16-bit file...
IF !doConvert=no GOTO 99

ConvertCommand=  !ConvertProg  !GenFlags  !Amplify  !SourceFlags  !SourceFile  !DestFlags
!MonoFile
EXECUTE !ConvertCommand
PAUSE

99


;------------

; ##  ###  ## ### #    #  # #
; # #  #   #  ### #   ###  #
; ##  ### ##  #   ### # #  #
```

```
100

FixedDisplayProg=no ; doesn't take command-line parameters yet

IF !FixedDisplayProg=no GOTO 199


ECHO display as pretty pictures
DisplayProg=!CDir !&sound.exe
ECHO The next program should display the waves...
EXECUTE !DisplayProg !MonoFile
199


;-------

;  #    #  #  #   # #  ## ###  ##
; ### ### ### #    #   #   #   #
; # # #   # # ###  #  ##  ### ##


200 ECHO C40 Octave Spectra Analysis

IF !doOsa=no ECHO skipping OSA...
IF !doOsa=no GOTO 259


IF !Machine=Wendy GOTO 210
201 ECHO This ONLY works on Wendy, the PC with the C40s attached.
END


210
PAUSE
EXECUTE !C40LoaderProg !C40Flags !AnalysisProg -i!MonoFile -o!SpectrumFile -m
PAUSE
259

;---------

IF !doDispSpec=no ECHO skipping spectrum display...
IF !doDispSpec=no GOTO 270


ECHO Now we'll try to display the spectra. The default parameters are for
ECHO Mendelssohn - tweak them for other tasks.
ECHO Press a key after the beep.
PAUSE
EXECUTE !DispSpecProg !SpectrumFile
PAUSE
270


;---------

IF !doCharacter=no ECHO skipping spectrum characterisation...
IF !doCharacter=no GOTO 280

ECHO The next program will look at this file to determine its average
ECHO power level. This will help later programs pick appropriate
ECHO thresholds.


ECHO Press a key after the beep.
PAUSE
EXECUTE !CharacterProg !SpectrumFile !CharacterFile
PAUSE
280


;---------

304 ; There's no special reason line numbers have to be in order
303 ; but it would seem to be a good idea on the whole.
302 ; Duplicate line numbers ARE trapped before the file is run,
```

```
301 ; but GOTO a nonexistent line number is only trapped at 'run-time'.


; ###  #   ## ###     ### ##   #   ## ###  ##  ## ###   # ##
; ### # #   #   #  ### ### ##  # # #   ##   #   #   #  ### # #
; #    #  ##   #      #   # #  #   ## ### ##  ##  ### #   ###


300 ECHO post-processing

IF !PostProcMethod=old GOTO 350


ECHO using NEW post-processing method - writing packed file
; remove sines
PickoutProg = !CDir !&pickout.exe
SinePackedListA = !WDir !& mex.spa
TestFlags = -o10 -m20
PickoutFlags = !SpectrumFile -g -p!SinePackedListA -s!SampleRate !TestFlags
ECHO RUNNING THIS MAY BE RATHER POINTLESS :)
PAUSE
EXECUTE !PickoutProg !PickoutFlags
345
ECHO haven't rewritten anything for further processing of PackedSineListA
END

;-------

350

IF !doPickout=no ECHO Skipping pickout...
IF !doPickout=no GOTO 359


ECHO using OLD method - writing text file

TestFlags = -o1 -m6 ;;; this was used until 17 august 94
TestFlags = -m6 -z24; more chance to extract all


TestFlags = -m6 -z48;;;;;;; try with wmendel

; 48 is too high
TestFlags = -m6 -z24 ;bells


PickoutFlags = !SpectrumFile -g -s!SampleRate !TestFlags -k!CharacterFile

; FILES WITH FFT <> 64 ?
IF !Task=schum PickoutFlags = !PickoutFlags -n32


ECHO !PickoutProg !PickoutFlags -t!SineListA
PAUSE

EXECUTE !PickoutProg !PickoutFlags -t!SineListA
PAUSE

359


;------

; The next bit won't work with packed lists.

400

IF !doA2B=no ECHO Skipping A -> B...
IF !doA2B=no GOTO 499


ECHO A to B
ECHO The next step is to sort the list of sines according to start time
ECHO This now uses MegaSort, which uses virtual memory.

PAUSE

DOS !A2BProg !SineListA !SineListB !A2BFlags
```

```
ECHO If this ran ok, we shouldn't need the file !SineListA
DOS DEL !SineListA /p
DOS CD ..

499


;------
500

IF !doShowSlb=no ECHO Skipping ShowSlb...
IF !doShowSlb=no GOTO 509


ECHO Now let's plot mex.slb to show how feasible it is...
ECHO After the beep, optionally capture the screen, then press a key.

ECHO !ShowSlbProg
PAUSE

EXECUTE !ShowSlbProg
PAUSE

509

;------


650
IF !doTrack=no ECHO Skipping tracking...
IF !doTrack=no GOTO 699

ECHO Track Sines ( !TrackProg !SineListB !ChainFile )
PAUSE
EXECUTE !TrackProg !SineListB !ChainFile
PAUSE
699

;-----

700
IF !doReorder=no ECHO Skipping reordering...
IF !doReorder=no GOTO 759

ECHO Reorder
ECHO
ECHO This needs as much memory as possible.
ECHO             Let's see how much you have here...
DOS mem
DOS mem >__before.cz
PAUSE
EXECUTE !ReorderProg !SineListB !ChainFile !ReorderedList
PAUSE
ECHO The amount of XMS memory should be the same as before
DOS mem
DOS mem >__after.cz
ECHO
ECHO If these files are different, then something went wrong.
DOS FC __before.cz __after.cz
PAUSE

759

;------
760

IF !doShowReo=no ECHO Skipping ShowReo...
IF !doShowReo=no GOTO 769


ECHO Now let's plot mex.Reo...
ECHO mex.reo is the reordered list
ECHO After the beep, optionally capture the screen, then press a key.

ECHO !ShowReoProg
PAUSE

EXECUTE !ShowReoProg
```

```
;;;;; ECHO      nice colours eh?
PAUSE

769

;------

800
IF !doFT=no ECHO Skipping frequency tracking...
IF !doFT=no GOTO 899


ECHO Frequency tracking
PAUSE
EXECUTE !FTProg !ReorderedList !TempDumpName
ECHO If this ran ok, you won't need the following file any more.
DOS DEL !ReorderedList /p
PAUSE

899

;---------

900
IF !doBattle=no ECHO Skipping battle...
IF !doBattle=no GOTO 999

DOS mem

PAUSE
ECHO Battle


EXECUTE !BattleProg !TempDumpName !AsciiFile -s!SampleRate -q
DOS DEL !TempDumpName !& .* /p

PAUSE


999

;-----------


1100

IF !doMakeWork=no GOTO 1150

ECHO Convert ASCII file to Cakewalk work file
DOS !Asc2WrkProg !AsciiFile !WorkFile
DOS COPY !WorkFile !WorkEditFile
PAUSE


1150

IF !doMakeMidi=no GOTO 1199

ECHO Convert Cakewalk work file to MIDI file
EXECUTE !Cake2MidProg !WorkFile !MidiFile

PAUSE

1199

;------

1200
IF !doCakewalk=no GOTO 1299

ECHO Start Cakewalk
DOS CD c:\Cakewalk
EXECUTE !Cakewalk !CakeFlags !WorkEditFile
DOS CD c:\c

PAUSE
```

```
1299

;-----

1300


ECHO that's all folks .'
;
; note - don't use an exclamation mark in a comment or ECHO
; cause the preprocessor thinks it's a variable name
;

END
```

10.7 Appendix G - Heisenberg's principle

This derivation is largely taken from Solbach.[Solbach]

Let $s(t)$ be a band-limited signal satisfying $\lim_{|t| \to \infty} s(t) \cdot \sqrt{t} = 0$

Given the energy of the signal $E = \int (|s(t)|)^2 dt = \int (|S(f)|)^2 df$

we define the time centre as $t_0 = \frac{1}{E} \cdot \int t \cdot (|s(t)|)^2 dt$

and the frequency centre as $f_0 = \frac{1}{E} \cdot \int f \cdot (|S(f)|)^2 df$

The time width is defined as $\Delta t_s = \sqrt{\frac{1}{E} \cdot \int (t - t_0)^2 \cdot (|s(t)|)^2 dt}$

and the frequency width is $\Delta f_s = \sqrt{\frac{1}{E} \cdot \int (f - f_0)^2 \cdot (|S(f)|)^2 df}$

Heisenberg's uncertainty[Papoulis] is then $\Delta t_s \cdot \Delta f_s \geq \frac{1}{4 \cdot \pi}$

10.8 Appendix H – Glossary of musical terms

additive synthesis	1) A synthesis technique where many components are added. Compare subtractive synthesis. 2) A synthesis technique based on generating a large number of sinusoids. Also called Additive Sine Wave Synthesis.
ADSR	Attack-Decay-Sustain-Release – a four-segment approximation to an amplitude envelope, commonly used on synthesisers.
aerophone	An instrument using a vibrating air column, including wind and brass instruments.
aftertouch	A feature of many synthesisers, where MIDI control messages are generated by pressing the key while holding it down. Aftertouch is implemented as either 'key aftertouch' for each note, or 'channel aftertouch', which is generated by any key and applies to the whole MIDI channel.
aliasing	A usually undesirable phenomenon in digital sampling, when a frequency above the Nyquist rate is reflected below it.
altissimo	The highest register of the clarinet, from Bb5 upwards, using the third mode of vibration, which is five times the fundamental frequency of the pipe.
alto	1) The (vocal) register below soprano, from A3 to D5. Alto is a contraction of contralto. 2) An alto instrument, such as the G alto flute, Eb alto clarinet. Eb alto saxophone, and Eb alto trombone.
amplitude	The size of an oscillation.
amplitude modulation	A simple synthesis technique that allows three partials to be computed for the computation cost of two.
anechoic	Listening conditions without echoes.
anharmonic	Same as inharmonic (2). (q.v.)
arpeggio	A chord played with a distinct gap between successive notes. Typically the lowest notes are played earliest.
artefact	An unwanted sound or distortion, often caused by non-linear editing operations.
attack	The starting transient of a note, often distinguished by high-frequency energy and instrument noise. Also see ADSR.
augmented	1) An interval one semitone larger than a perfect unison, fourth, fifth, or octave. 2) A chord containing a major third and an augmented fifth.
aural harmonic	A harmonic cause by non-linear transmission in the ear.
bar	1) A regular grouping of several beats. (U.S. meter) 2) The wood/metal bar of a percussion instrument. The bar has inharmonic modes of vibration.
baritone	1) The (vocal) register between tenor and bass. 2) A baritone instrument such as the Eb baritone saxophone. 3) A brass instrument similar to a euphonium but with a narrower bore. (U.S. baritone horn)
bark	A unit of subjective pitch defined as the critical bandwidth. One bark is approximately equal to 100 mels.
bass	1) The (vocal) register below baritone, from F2 to D4. 2) A bass instrument such as the D bass flute, Bb/A bass clarinet, Bb bass trumpet, Bb bass saxophone, and Eb/F bass tuba. 2) Common term for bass guitar, double bass, or tuba.
BBb	'Double B (flat)' is a colloquial term referring to the pitch of a contrabass tuba. It is also occasionally applied to the bass saxophone, or contrabass clarinet.
beat	The fundamental short unit of time, typically equivalent to a quaver, a crotchet, or a minim.

bel	Ten decibels; the unit Bel is rarely seen on its own.
bell	1) A church bell, hand bell, or tubular bell. 2) The expansion in bore at the end of a brass or wind instrument.
binaural	Sound presented to both ears. This is usually stereo, but may also be mono.
bore	The width profile, or the width in a cylindrical section, of a brass or wind instrument.
bpm	Abbreviation for beats per minute. See tempo.
brass	A subcategory of aerophone based on vibrating lips, also known as lip reed instruments. (U.S. brasswind) Not all brass instruments are made of brass, as exemplified by the fibreglass sousaphone, didgeridoo, conch shell, and serpent.
breve	A note length of two semibreves or eight crotchets. Breves are rarely used now, yet they represented the shortest note in medieval times.
CD-quality	Loose term for recording or playback with 16-bit linear encoding, a 44100-Hz sample rate, and two channels.
cent	A frequency ratio of $\sqrt[1200]{2}$. A hundred cents make one semitone.
chalumeau	The lowest register of the clarinet, from D3 to E4, using the fundamental mode of vibration of the pipe.
channel	The MIDI standard allows 16 channels to represent 16 instruments.
chord	Several notes played nearly simultaneously. In practice unintentional asynchrony is on the order of 10-20 ms.
chordophone	An instrument based on a vibrating string, whether bowed (violin), plucked (harp), or struck (piano).
chromatic	A sequence of semitone steps.
circle of fifths	The chromatic scale reordered modulo 7; C-G-D-A-E-B-Gb-Db-Ab-Eb-Bb-F-C.
clarion	The middle register of the clarinet, from A4 to B5, using the second mode of vibration, which is three times the fundamental frequency of the pipe.
clef	The CPN symbol indicating the pitch range of a staff. Treble and bass clefs are common; alto is used for violas and tenor is used for trombones and bassoons. The soprano, mezzo-soprano, and baritone clefs have fallen into disuse.
clipping	Distorting a signal by peak-limiting.
cocktail party effect	Our ability to distinguish one auditory stream (especially speech) out of many.
combination tone	A tone heard at the sum of two frequencies.
common practice notation (CPN)	The conventional method of notating music by drawing notes and other symbols on staves.
compensation	A system for correcting mistuning on (low) valved brass instruments.
compression	1) Reducing the size of a digital file by recoding and possibly simplifying. 2) A method of limiting the amplitude range by boosting quiet sounds and attenuating loud sounds.
consonance	The perceptual pleasantness of two or more simultaneous tones.
contrabass	1) The register below bass. 2) A double bass, also known as a string bass. 3) A contrabass instrument such as the Bb contrabass clarinet or saxophone, or the BBb/CC contrabass tuba.
contralto	See alto.
critical bandwidth	The 'interval' below which two pure tones will interfere. See masking.
crotchet	The 'basic' unit of musical time. (U.S. quarter-note)
DarthVaderisation	Colloquial term for the perceptual change in timbre when a wave is played back slower than intended. Opposite of the Mickey Mouse effect.

decay	1) see ADSR 2) The reverberation time of a real acoustic environment.
delay	1) Computational delay in processing. 2) A common analogue and digital effect, where a delayed copy of a signal is added to the original.
difference tone	A tone heard at the difference between two frequencies.
diminished	1) An interval one semitone smaller than a perfect unison, fourth, fifth, or octave. 2) A chord containing a minor third and a diminished fifth.
dissonance	Opposite of consonance.
distortion	1) Undesired noise or deviations in a signal. 2) An intentional processing effect, popular with electric guitars.
Dolby noise reduction	A proprietary system for reducing tape hiss by boosting high frequencies recorded and attenuating them on playback.
double stop	Two notes played on neighbouring strings of a bowed string instrument.
driven	In instrument hierarchies, driven instruments are those that are excited continuously, such as brass/wind instruments and bowed strings. Non-driven instruments are excited by a short high-energy impulse, and include plucked strings, pianos, and most percussion instruments.
duration	1) The actual length of a note (when the start and end can be defined). 2) The length in ticks of a MIDI note. 3) The written duration of a note in CPN, e.g. dotted minim.
dynamic range	The difference between the loudest sounds and the quietest, on an audio system or musical instrument.
effect	Any analogue or digital processing of a signal, including reverberation, phasing, flanging, equalisation, and compression.
enharmonic	Two different names for a single note in equal temperament, e.g. F# and Gb. (Not the same as inharmonic.)
envelope	This term usually refers to the amplitude of a harmonic as a function of time – typically approximated by line segments. It can also refer to frequency profiles or any other continuous control parameter.
equal temperament (ET)	A type of temperament (q.v.) in which (usually) 12 notes per octave are spaced equally.
equalisation	Adjusting the sound by filtering to give better overall spectral balance.
flanging	A distortive effect using modulation of the delay time in a delay unit.
flutter	An undesirable effect on record and tape decks caused by a varying playback rate.
flutter tonguing	An effect on brass instruments played using a strongly rolled 'r'.
formant	A strong frequency band that is independent of pitch on an instrument, particularly the human voice.
frequency modulation	1) For carrier frequencies around 5-10 Hz, a method of simulating vibrato. 2) For higher carrier frequencies, a common and computationally modest way to achieve a rich spectrum. See also phase modulation.
frequency shifting	1) Literally, the operation of moving frequencies up or down by a constant frequency, as achieved by single-sideband modulation. Musically, this is undesirable. 2) Common but strictly inaccurate term for pitch shifting. (q.v.)
frequency stretching	1) An effect of instrument non-linearity wherein the frequencies of the partials increase faster than the partial number. This often reflects non-ideal behaviour of strings (bass piano strings), bars, or pipes (organ, bell-less brass). 2) Loose term for Railsback stretch (q.v.)

fundamental	The frequency implied by the spectrum of a note, generally the lowest frequency. When the note has a harmonic spectrum, this frequency corresponds to the repetition period of the waveform.
General MIDI (GM)	A superset of MIDI that also defines a set of standard instruments.
glissando	A smooth transition in pitch. Examples include voice, trombone, and fretless strings.
grace note	A short note preceding a main note, possibly a few semitones away.
half-valve	To play a brass instrument with one or more valves partially depressed.
hard left/right	The extremes of pan position.
harmonic	The n^{th} harmonic is nominally the component at n times the fundamental frequency.
harmony	The spectral context formed by chords and notes.
horn	1) Loose term for the French horn, or sometimes the Eb tenor horn (U.S. alto horn). 2) U.S. slang term for all brass instruments *and* saxophones.
hypersonic	A frequency above 20 kHz.
idiophone	An instrument based on a solid resonator, such as a xylophone, cymbal, or cowbell.
IID	Interaural intensity difference, the ratio of the sound reaching both ears.
inflection	Any acoustic information not represented by the musical score. This includes time inflections (rush/drag), note inflections (ornaments), pitch inflections (vibrato, sharpness/flatness), amplitude inflections (tremolo) and timbral inflections (filtering).
infrasonic	A frequency below the normal hearing range, i.e. below 20 Hz.
inharmonic	1) An interval not corresponding closely to a simple frequency ratio. 2) A spectrum where the frequency peaks are not integrally related.
interval	1) Pitch/frequency interval – The logarithm of the ratio between two frequencies or pitches. This can be expressed in semitones or by a mode-specific term (major/minor second/third/sixth/seventh, diminished/perfect/augmented unison/fourth/fifth/octave). 2) Time interval – see IOI.
inversion	The position of the bass with respect to the root of a chord.
IOI	Inter-Onset Interval, the length of time between the onsets of consecutive notes.
ITD	Interaural time difference, the time between a sound reaching the closer ear and the further.
just noticeable difference (JND)	The resolution, or smallest perceptible variation, of a quantity such as frequency or amplitude.
just temperament (just intonation)	A temperament where simpler intervals are adjusted to integral ratios at the expense of making other intervals less in-tune.
karaoke	Japanese term for 'empty orchestra', referring to a "Music Minus One" system for real-time accompaniment of a singer.
key	1) The long-term 'centre' of the harmony – often expressed as a pitch class and a mode – e.g. C major. 2) The key covering a hole on a woodwind or brass instrument.
legato	A style of note articulation where notes are connected.
lip trill	A type of trill on brass instruments played by rapid oscillation between two modes of vibration without using the valves.
loudness (S)	A pure tone with loudness level L_L has a loudness S in sones, given by $S = 2^{((L_L-40)/10)}$.
loudness level (L_L)	The loudness level of a tone is the sound pressure level of a 1000-Hz tone that is equally loud.

major	1) A scale in the Ionian mode, such as the white notes from C to C. 2) A chord containing a major third and a perfect fifth. 3) An interval such as a major second, third, sixth, or seventh, consisting of 2, 4, 9, or 11 semitones respectively.
masking	An auditory phenomenon by which one frequency 'drowns out' other close frequencies.
McGurk effect	The dependence of auditory perception on simultaneous visual cues.
mel	A little-used unit of subjective pitch that takes into account poor low-frequency discrimination.
membrane	The circular head of a drum. Membranes have inharmonic modes of vibration.
membranophone	An instrument based on a vibrating membrane – i.e. a drum.
Mickey Mouse effect	Colloquial term for the perceptual change in timbre when a wave (especially a voice) is played faster than the correct speed.
micropascal (μPa)	A small unit of pressure, one millionth of a Pascal.
MIDI	Musical Instrument Digital Interface, the current standard communication protocol for synthesisers and computers.
minim	Equal to two crotchets. (U.S. half-note)
minor	1) A scale in the Aeolian mode, such as the white notes from A to A. 2) A chord containing a minor third and a perfect fifth. 3) An interval such as a minor second, third, sixth, or seventh, consisting of 1, 3, 8, or 10 semitones respectively.
missing fundamental	See residue tone.
mode	1) An attribute showing which note of a 7-note scale (q.v.) is the root note. There are thus 7 modes, and their classical names are Ionian (major), Dorian, Phrygian, Lydian, Mixolydian, Aeolian (minor), and Locrian. 2) A mode of vibration of an instrument.
modulation	1) Varying one signal using another, as in amplitude, frequency, or phase modulation (q.v.) 2) A change of key, often to a closely related key.
mono (monaural)	Sound from a single audio output channel.
monophonic	A musical sound consisting of at most one note. See polyphony.
monotimbral	Sound with a single timbre, possibly polyphonic.
mordent	A motif of 2 or 3 grace notes.
multiphonic	Two or more notes played simultaneously on a normally monophonic instrument. Flutes, saxophones, horns, and trombones are capable of playing multiphonics.
munchinization	Another term for the Mickey Mouse effect.
music minus one	Term by Irv Kratka for accompaniment without the soloist. MMO systems include karaoke systems.
mute	A device added to a string or brass instrument to vary the timbre.
noise	Any unwanted component in the sound.
note	The musical equivalent of a syllable.
Nyquist frequency ***Nyquist rate***	Half the sample rate of a digitally encoded signal, e.g. 22050 Hz (CD), 24000 Hz (DAT).
octave	Physically, a frequency ratio of 2. The ratio that is judged to be perceptually an octave is very slightly larger.
offset	The 'end time' of a note, as in the time of key release.
onset	The start of a note, where one can be defined.

open	A note played on a brass instrument with no valves down, or on a string instrument without left-hand fingering.
Open Window Unit	A measure of sound absorption. One person in a concert hall absorbs the amount of energy that would be lost through an open window 0.44 m^2 in size, so one person has an absorption of 0.44 Open Window Units.
ornament	Any type of note modulation, including grace notes, mordents, and trills.
overblowing	Moving between modes of vibration where the instrument's acoustic length does not change, as on a wind or brass instrument.
overtone	The n^{th} overtone of a note is the $(n+1)^{th}$ partial. This term is confusing and is thus usually avoided.
pan (position)	The perceptual position of a sound on the left-right axis.
partial	A component frequency above the fundamental. When the nth partial is at or very near n times the fundamental frequency, the term 'harmonic' is used.
Pascal	A unit of pressure. 1 Pa = 1 N/m^2 = 10 dynes/cm^2 = 10 microbars.
patch	On synthesisers, the parameter settings that map a MIDI note to the actual sound.
pentatonic	A five-note scale, a transposed version of C-D-E-G-A.
percussion	1) An instrument excited by striking, sometimes taken to include pianos. 2) Any instrument not played by the rest of the orchestra. This includes most unpitched instruments and some pitched instruments including bells, timpani, and whistles.
perfect	An interval such as a perfect unison, fourth, fifth, or octave consisting of 0, 5, 7, or 12 semitones.
phase modulation	A very similar technique to frequency modulation, with similar results. Many "FM" synthesisers and soundcards actually implement the phase modulation algorithm.
phasing	A similar effect to flanging (q.v.) but at higher modulation frequencies.
phon	A little-used and somewhat arbitrary unit of perceptual loudness.
pitch	The term 'pitch' may refer to physical pitch or perceptual pitch. Physical pitch is the logarithm of the fundamental frequency, usually expressed in semitones from a fixed standard. The prevailing standard defines the A above middle C to be 440 Hz.
pitch bend	Pitch modulation, or a controller for it.
pitch chroma ***pitch class***	The twelve pitch classes, or chroma, – C, C#, D, ..., A#, B – represent the perceptually similar notes across octaves.
pitch height	The dimension defining high notes and low notes, as distinguished from pitch class.
pitch modulation	Any variation in the pitch of a note from the notated pitch. Periodic variation includes vibrato.
pitch shifting	Processing a signal so as to change its pitch profile while retaining the same time profile. This is essentially the same procedure as time stretching followed by resampling.
plate	A metal percussion instrument that is essentially flat, such as a cymbal or a gong. It usually excludes bar instruments (q.v.).
polyphony	The number of simultaneous notes.
polytimbrality	The number of simultaneous instruments.
portamento	A short glissando (q.v.) played between notes.
quadraphonic	Sound presented through four channels.
quaver	Half a crotchet. (U.S. eighth-note)
Railsback stretch	A phenomenon where the tuning of an instrument is sharper for higher frequencies, as in a piano. This is related both to frequency stretching and to other non-linearities.
range	The range of pitches that can be played by a given instrument.
reed	The reed of wind instrument, either a single reed, as in a clarinet or saxophone, or a double reed, as in the oboe/bassoon family.

register	1) A range of pitches spanning around 1½ octaves (corresponding to the range of human voices). From highest to lowest, the overlapping ranges are (sopranino), soprano, alto, tenor, (baritone), bass, (sub-bass), contrabass, (sub-contrabass), the terms in parentheses being less common. These terms also distinguish different instruments in the same family. 2) A specific mode of vibration on a woodwind instrument.
release	see ADSR
resampling	Converting a waveform from one sample rate to another.
residue tone	A harmonic note with no energy at the fundamental frequency. The perceived pitch is that of the missing fundamental.
reverberation	Natural or artificial repetitions of the source signal.
rhythm	A repetitive motif in time.
ring modulation	A relative of amplitude modulation in which the carrier frequency is not present, also called balanced modulation and double-sideband suppressed carrier modulation.
rip	A similar effect to portamento on a brass instrument where the player moves rapidly up through the harmonic series.
roll	A fairly even series of strikes on a membrane, plate, or bar. Typically these are not written individually.
root	The pitch class of a chord.
rubato	Italian for "robbed time", i.e. playing with flexible and expressive musical timing.
rumble	Low-frequency noise, especially on a record deck.
sample	1) A digital recording of a wave lasting typically 1-30 seconds. 2) A recording of a single note. 3) A single point in a digitised wave.
savart	A little-used pitch unit of a three-hundredth of an octave, or four cents.
saxhorn	The family of conical-bore brass instruments invented by Adolphe Sax, including flugel horns, tenor horns, baritones, euphoniums, and tubas.
scale	The set of notes chosen from the twelve pitch classes. Commonly there are 7, and the spacings are 2-2-1-2-2-2-1. See also mode.
scientific pitch	A computationally convenient but rather flat tuning scheme based on middle C=256 Hz (and possibly a sample rate of a power of two). In any case, the semitone is irrational.
semibreve	Equal to four crotchets. (U.S. whole note)
semiquaver	A quarter of a crotchet. An eighth and a sixteenth of a crotchet are called a demisemiquaver and a hemidemisemiquaver. (U.S. sixteenth-note, 32nd-note, 64th-note)
semitone	A twelfth of an octave – thus a frequency ratio of $\sqrt[12]{2} \approx 1.05946$. A calculator-friendly approximation is $\sqrt[4]{(1.26)}$. Occasionally called semit.
slur	A line drawn connecting notes to be played legato, or notes thus played.
stereo	Sound presented through two channels.
Stevens's Rule	The phenomenon whereby tones below 2 kHz are heard as flatter, and tones above as sharper, as intensity increases. S. S. Stevens described this in 1935.
solfa / solfeggio	The little-used scale of do-re-mi-fa-so-la-te-do. There are other syllables to form the complete twelve-note scale.
sone	A unit of loudness. The number of sones is given by $2^{((L_L-40)/10)}$ where L_L is the loudness level.
sopranino	1) The register above soprano. 2) A sopranino instrument such as the Eb sopranino saxophone.

soprano	1) A high (vocal) register, above alto, from C4 to G5. 2) A soprano instrument such as an Eb soprano cornet, Eb soprano clarinet, and Bb soprano saxophone.
sound intensity level (IL, L_I)	The rate of energy flow per unit area, relative to 10^{-12} watt/m^2.
sound power level (PWL, L_W)	The total power in all directions, relative to one picowatt (10^{-12} W).
sound pressure level (SPL, L_p)	The pressure of a sound, relative to 20 micropascals.
speed of sound	The rate of sound propagation. In air, this is approximately 331.3 + 0.6×T m/s, where T is the temperature.
staccato	A style of playing where notes are very short, or a CPN marking to indicate this.
staff ***stave***	The set of five lines in Common Practice Notation (q.v.).
sub-bass	A clef one octave below the bass clef, proposed by Rossing, for the contrabass register. It is indicated by two bass clefs.
subcontrabass	The ultra-low register below contrabass, or an instrument in this range. Such instruments are rare; the least rare being the subcontrabass clarinet. Other terms are octo-contrabass and octobass.
subtractive synthesis	A synthesis technique wherein a high-bandwidth signal, such as noise or impulses, is filtered into the desired sound.
summation tone	Same as a combination tone.
super-treble	A clef one octave above the treble clef, proposed by Rossing and shown by two treble clefs.
supra-super-treble	A clef two octaves above the treble clef, proposed by Rossing and shown by three treble clefs.
surround sound	A type of quadraphonic sound system.
sustain	see ADSR
system	In printed music, a set of simultaneous staves.
temperament	The precise tuning of the twelve pitch classes. This tuning usually repeats every octave. See equal temperament and just temperament.
tempo	The perceived speed of the piece, typically expressed in crotchets per minute or beats per minute. Tempi implemented by a metronome range from 30 to 240 bpm, corresponding to frequencies from ½ to 4 Hz.
tenor	1) The (vocal) register between alto and baritone, from C3 to G4. 2) A tenor instrument, such as the Eb tenor horn, Bb tenor trombone, and the Bb tenor saxophone.
threshold of audibility	The minimum pressure fluctuation that can be heard. Typically this is 20 µPa at 1000 Hz.
tick	The clock frequency of MIDI equipment, typically 96 or 120 ppqn (pulses per quarter-note)
tie	A line connecting separately notated parts of a single note.
timbre	The complex attribute that distinguishes notes with the same fundamental frequencies, loudnesses, and spatial locations.
time	1) Physical time, in milliseconds or hours. 2) Musical time in terms of bars and beats.
time shifting ***time stretching*** ***timescale modification***	Processing a wave with the intention of changing its time profile while leaving its frequency profile unaltered.

tone	1) A near-harmonic note, usually with several components. 2) "Twelve-tone" music refers to 12 pitch classes (q.v.) 3) Two semitones, equal to a ratio of $^{6}\sqrt{2}$ in equal temperament. 4) Loose term for timbre, or specific attributes of timbre.
tongue	To clearly articulate the start of a note on a brass or woodwind instrument.
transcribe	1) To write down music in CPN. 2) To rearrange music for different instrumentation.
transient	1) Inharmonic components at the start of a note, often associated with the excitation mechanism. 2) Unusual sounds produced between notes. These may be due to the instrument changing shape (moving a key/valve/finger), changing mode of vibration (overblowing), or noise from mechanical parts.
transpose	To change the key of a piece of music.
treble	1) The higher of the two standard clefs in CPN. 2) Loose term for high-frequency content.
tremolo	An oscillation in amplitude, usually of 2-10 Hz.
trill	A rapid alternation between two notes.
tritone	Half an octave, i.e. six semitones. In equal temperament, equivalent to an augmented fourth or a diminished fifth.
vibrato	An oscillation in frequency, usually of 2-10 Hz.
virtual pitch	The pitch of a residue tone (q.v.). Also known as residue pitch, low pitch, periodicity pitch, time-separation pitch, and repetition pitch.
volume	1) The amplification setting of a music playback device. 2) Loose term for loudness.
wind	An aerophone. The broadest definition is woodwind, voice, and brass; a narrower definition is woodwind and voice only.
woodwind	An aerophone not including brass instruments or the voice. Non-wooden woodwind include flutes, saxophones, and contrabassoons.
wow	An undesirable effect in record and tape decks, similar to flutter (q.v.), but at lower frequencies.
ZIPI	A communications standard designed to supersede MIDI (q.v.).

Table 45 - Glossary of musical terms.

10.9 Appendix I – Acronyms

ADPCM	Adaptive Differential Pulse Code Modulation
ADSR	Attack-Decay-Sustain-Release
AM	Amplitude Modulation
ASIC	Application-Specific Integrated Circuit
ASWS	Additive Sine Wave Synthesis
BGI	Borland Graphics Interface
CASA	Computational Auditory Scene Analysis
CPN	Common Practice Notation
DFT	Discrete Fourier Transform
DOS	Disk Operating System
DPCM	Differential Pulse Code Modulation
DSP	Digital Signal Processing
DWT	Discrete Wavelet Transform
ET	Equal Temperament
FFT	Fast Fourier Transform
FIR	Finite Impulse Response
FM	Frequency Modulation
FT	Fourier Transform
GAS	Group Additive Synthesis
GFT	Generalised Fourier Transform
GIF	Graphics Interchange Format
GM	General Midi
GUI	Graphical User Interface
GUS	Gravis UltraSound
HRTF	Head-Related Transfer Function
IID	Interaural Intensity Difference
IIR	Infinite Impulse Response
IOI	Inter-Onset Interval
ITD	Interaural Time Difference
JND	Just-Noticeable Difference
JPEG	Joint Photographic Experts Group
MFT	Multiresolution Fourier Transform
MIDI	Musical Instrument Digital Interface
MIMD	Multiple-Instruction Multiple-Data
MPEG	Motion Pictures Experts Group
PCM	Pulse Code Modulation
PLA	Piecewise-Linear Approximation
PM	Phase Modulation *or* Physical Modelling
QMF	Quadrature Mirror Filter
RM	Ring Modulation
SIMD	Single-Instruction Multiple-Data
SNR	Signal-to-Noise Ratio
STFT	Short-Time Fourier Transform
SVGA	Super VGA
TSR	Terminate and Stay Resident
VESA	Video Electronics Standards Association
VGA	Video Graphics Adapter
VLSI	Very Large-Scale Integration
WT	Wavelet Transform

Table 46 - Acronyms and abbreviations.

10.10 Appendix J – Terms I define

atom	A representation of an array of quanta.
compact (species)	A species of atom other than species 15.
convolentiation	Convolentiation is to convolution as exponentiation is to multiplication.
density (of a quantum)	The factor α in the Gaussian $e^{-\alpha t^2}$.
inside-out	The Fourier Transform of an atom.
molecule	An array of atoms.
quantum	A Gabor wavelet.
species	A number (0-15) describing the data format of the four arrays in an atom.

Table 47 - Terms I define in this thesis.

10.11 Appendix K – Analysis of Träumerei

This experiment was carried out to repeat some of the analyses carried out by Repp.[Repp, Widmer] The first stage is to determine the input times using a Windows sound file editor. Some times can be determined to a high accuracy, but others in note clusters have more variance. From the raw times, we can derive the inter-onset intervals (IOIs), and from there the variation in tempo with time. Only the first four bars were used.

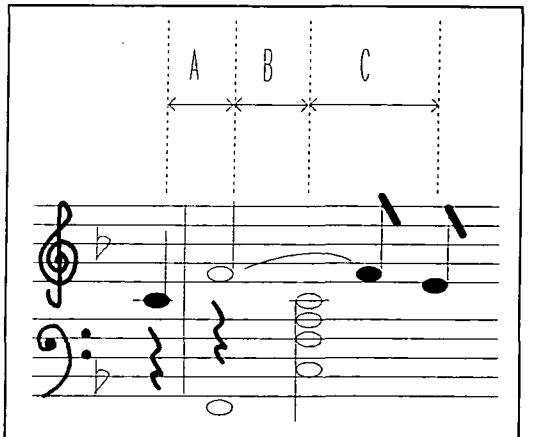


Figure 152 - Melodic gesture MG1.

Repp further classifies the performances by examining 'melodic gestures'. The first, MG1, uses the first three IOIs – C-F, F-chord, and chord-E – and labels these times A, B, and C, as shown in Figure 152.

He calculates the values of A/(B+C) and B/C for 24 of the 28 pianists, normalised using the 'literal' values. The performance analysed gave 1.03 and 1.46 for these values respectively, showing that the performer played the upbeat literally but delayed the chord. By comparing this to the first panel of Figure 5 of Repp's paper, this value is closest to the performance by ARR (Claudio Arrau) or possibly DEM (Jörg Demus).

The next gesture to be analysed is the run of five quavers shown in Figure 153. As illustrated, the IOIs tend to follow a parabolic function, with the quaver length decreasing then increasing over the phrase. The curvature was derived by fitting a curve using MathCad, giving a value of Q=100.[90] From Figure 9 of Repp's paper, this value corresponds to Arrau or a few others, but not to Demus, whose 'expression' parabola is even more curved.

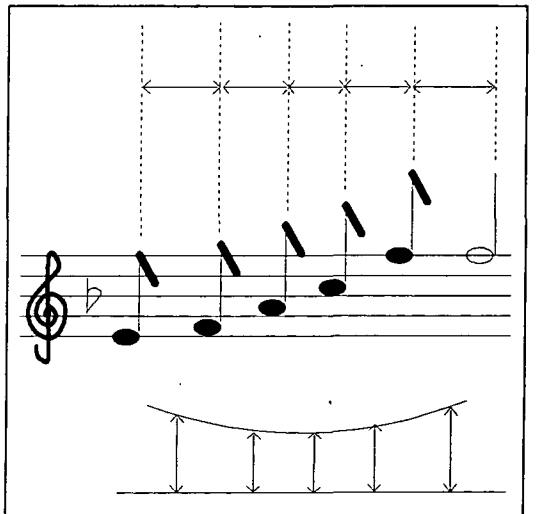


Figure 153 - Melodic gesture MG2.

[90] The units of this curvature are milliseconds per cubic quaver, dimensions even sillier than those for 'Stevens's constant' of 3.362×10^{-4} ¢.dB^{-1}.Hz^{-1} in Chapter 2.

Repp next analyses the remaining quavers. The input stops on the G minim in bar 4, so this analysis could only evaluate gestures MG3a/ MG4a/MG5a (notated in Figure 154 differently to the score), but not the quavers in the bass line (MG6a). Repp uses principal components analysis to derive six timing patterns. The IOI graph derived for the unknown pianist is closest to that for Factor I, with which Arrau has a high correlation. It was thus deduced, correctly, that the pianist was Claudio Arrau.

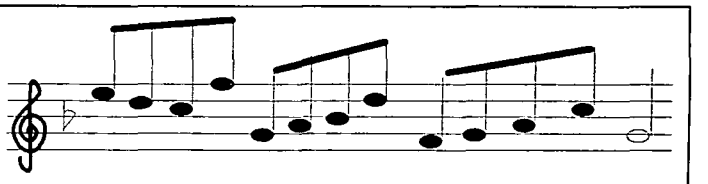


Figure 154 - Melodic gestures MG3a, MG4a, and MG5a.

10.12 Appendix L – Mendelssohn Csound files

The score file for creating the Mendelssohn example is shown below.

```
w 0 68
f 1 0 0 512 512 10 15 10 6 9 4 5 2 8 3 2 1 3
f 2 0 0 513 513 5 0.001 512 1
f 3 0 0 512 512 10 15 12 8 9 4 5 3 4
f 0 1 .882353
s
w 0 60
i 1   0    0    0.5  0.5  9.04 3400 0.02 0.03
i 2   0    0    0.5  0.5  9.01 3400 0.02 0.03
i 3   0    0    1    1    8.09 3400 0.02 0.03
i 4   0    0    0.5  0.5  8.01 3400 0.02 0.03
i 5   0    0    0.5  0.5  7.09 3400 0.02 0.03
i 7   0    0    1    1    7.09 3400 0.02 0.03
i 8   0    0    1    1    6.09 3800 0.02 0.03
i 9   0    0   -1   -1
i 1   0.5  0.5  0.5  0.5  9.02 3400 0.02 0.03
i 2   0.5  0.5  0.5  0.5  8.11 3400 0.02 0.03
i 4   0.5  0.5  0.5  0.5  8.02 3400 0.02 0.03
i 5   0.5  0.5  0.5  0.5  7.11 3400 0.02 0.03
i 1   1    1    0.5  0.5  9.01 3400 0.02 0.03
i 2   1    1    0.5  0.5  8.09 3400 0.02 0.03
i 4   1    1    1    1    8.04 3400 0.02 0.03
i 5   1    1    0.5  0.5  8.01 3400 0.02 0.03
i 7   1    1    1    1    7.04 3400 0.02 0.03
i 8   1    1    1    1    6.04 3800 0.02 0.03
i 1   1.5  1.5  0.5  0.5  8.11 3400 0.02 0.03
i 2   1.5  1.5  0.5  0.5  8.08 3400 0.02 0.03
i 5   1.5  1.5  0.5  0.5  8.02 3400 0.02 0.03
i 1   2    2    1.5  1.5  8.09 3400 0.02 0.03
i 2   2    2    1.5  1.5  8.06 3400 0.02 0.03
i 4   2    2    1.5  1.5  7.09 3400 0.02 0.03
i 5   2    2    1.5  1.5  8.01 3400 0.02 0.03
i 7   2    2    1.5  1.5  7.06 3400 0.02 0.03
i 8   2    2    1.5  1.5  6.06 3800 0.02 0.03
i 1   3.5  3.5  0.5  0.5  8.09 3400 0.02 0.03
i 2   3.5  3.5  0.5  0.5  8.08 3400 0.02 0.03
i 4   3.5  3.5  0.5  0.5  8.01 3400 0.02 0.03
i 5   3.5  3.5  0.5  0.5  7.09 3400 0.02 0.03
i 7   3.5  3.5  0.5  0.5  7.04 3400 0.02 0.03
i 8   3.5  3.5  0.5  0.5  6.04 3800 0.02 0.03
i 1   4    4    1    1    9.06 3400 0.02 0.03
i 2   4    4    1    1    9.02 3400 0.02 0.03
i 3   4    4    1    1    8.09 3400 0.02 0.03
i 4   4    4    1    1    8.06 3400 0.02 0.03
i 5   4    4    1    1    8.06 3400 0.02 0.03
i 6   4    4    1    1    8.02 3400 0.02 0.03
i 7   4    4    1    1    7.02 3400 0.02 0.03
i 8   4    4    1    1    6.02 3800 0.02 0.03
i 1   5    5    0.75 0.75 9.04 3400 0.02 0.03
i 2   5    5    0.75 0.75 9.01 3400 0.02 0.03
i 3   5    5    1.5  1.5  8.09 3400 0.02 0.03
i 7   5    5    1    1    6.09 3400 0.02 0.03
i 8   5    5    1    1    5.09 3800 0.02 0.03
i 1   5.75 5.75 0.25 0.25 9.06 3400 0.02 0.03
i 2   5.75 5.75 0.25 0.25 9.02 3400 0.02 0.03
i 1   6    6    1    1    9.04 3400 0.02 0.03
i 2   6    6    1    1    9.01 3400 0.02 0.03
i 7   6    6    2    2    6.11 3400 0.02 0.03
i 8   6    6    2    2    5.11 3800 0.02 0.03
i 3   6.5  6.5  0.5  0.5  8.06 3400 0.02 0.03
i 1   7    7    0.5  0.5  9.02 3400 0.02 0.03
i 2   7    7    0.5  0.5  8.11 3400 0.02 0.03
i 1   7.5  7.5  3    3    8.11 3400 0.02 0.03
i 2   7.5  7.5  0.5  0.5  9.02 3400 0.02 0.03
i 3   7.5  7.5  0.5  0.5  8.09 3400 0.02 0.03
i 2   8    8    0.75 0.75 9.08 3400 0.02 0.03
i 3   8    8    1    1    8.08 3400 0.02 0.03
i 4   8    8    1    1    9.04 3400 0.02 0.03
i 7   8    8    2    2    6.04 3400 0.02 0.03
i 8   8    8    2    2    5.04 3800 0.02 0.03
i 2   8.75 8.75 0.12 0.12 9.06 3400 0.02 0.03
i 2   8.87 8.87 0.13 0.13 9.08 3400 0.02 0.03
i 2   9    9    0.5  0.5  9.09 3400 0.02 0.03
i 3   9    9    0.5  0.5  8.09 3400 0.02 0.03
i 4   9    9    0.5  0.5  8.06 3400 0.02 0.03
i 2   9.5  9.5  0.5  0.5  9.11 3400 0.02 0.03
i 3   9.5  9.5  1.5  1.5  8.08 3400 0.02 0.03
i 4   9.5  9.5  2.5  2.5  8.04 3400 0.02 0.03
i 2   10   10   2.5  2.5  9.04 3400 0.02 0.03
i 7   10   10   2    2    7.01 3400 0.02 0.03
i 8   10   10   2    2    6.01 3800 0.02 0.03
i 1   10.5 10.5 0.5  0.5  9.02 3400 0.02 0.03
i 1   11   11   1.5  1.5  9.01 3400 0.02 0.03
i 3   11   11   0.5  0.5  8.09 3400 0.02 0.03
i 3   11.5 11.5 0.5  0.5  8.08 3400 0.02 0.03
i 3   12   12   2    2    8.06 3400 0.02 0.03
i 7   12   12   1    1    7.02 3400 0.02 0.03
i 8   12   12   1    1    6.02 3800 0.02 0.03
i 1   12.5 12.5 0.5  0.5  8.09 3400 0.02 0.03
i 1   13   13   0.5  0.5  9.04 3400 0.02 0.03
i 2   13   13   0.5  0.5  9.01 3400 0.02 0.03
i 7   13   13   1    1    6.11 3400 0.02 0.03
i 8   13   13   1    1    5.11 3800 0.02 0.03
i 1   13.5 13.5 0.5  0.5  9.02 3400 0.02 0.03
i 2   13.5 13.5 0.5  0.5  8.11 3400 0.02 0.03
i 1   14   14   1    1    9.01 3400 0.02 0.03
i 2   14   14   1    1    8.09 3400 0.02 0.03
i 3   14   14   1.5  1.5  8.04 3400 0.02 0.03
i 7   14   14   7    7    7.04 3400 0.02 0.03
i 8   14   14   7    7    6.04 3800 0.02 0.03
i 1   15   15   0.5  0.5  8.11 3400 0.02 0.03
i 2   15   15   0.5  0.5  8.08 3400 0.02 0.03
i 1   15.5 15.5 0.5  0.5  8.09 3400 0.02 0.03
i 2   15.5 15.5 0.5  0.5  8.06 3400 0.02 0.03
i 3   15.5 15.5 1.5  1.5  7.11 3400 0.02 0.03
i 1   16   16   0.5  0.5  8.11 3400 0.02 0.03
i 2   16   16   0.5  0.5  8.08 3400 0.02 0.03
i 1   16.5 16.5 0.5  0.5  8.09 3400 0.02 0.03
i 2   16.5 16.5 0.5  0.5  8.06 3400 0.02 0.03
i 1   17   17   0.5  0.5  8.08 3400 0.02 0.03
i 2   17   17   0.5  0.5  8.04 3400 0.02 0.03
i 4   17   17   0.5  0.5  7.11 3400 0.02 0.03
i 5   17   17   1    1    6.11 3400 0.02 0.03
i 1   17.5 17.5 0.5  0.5  8.06 3400 0.02 0.03
i 2   17.5 17.5 0.5  0.5  8.03 3400 0.02 0.03
i 4   17.5 17.5 0.5  0.5  7.09 3400 0.02 0.03
i 1   18   18   1    1    8.04 3400 0.02 0.03
i 4   18   18   0.5  0.5  7.08 3400 0.02 0.03
i 5   18   18   0.5  0.5  7.11 3400 0.02 0.03
i 4   18.5 18.5 0.5  0.5  7.09 3400 0.02 0.03
i 1   19   19   2.5  2.5  8.11 3400 0.02 0.03
i 4   19   19   3    3    7.11 3400 0.02 0.03
i 5   19   19   0.5  0.5  7.08 3400 0.02 0.03
i 5   19.5 19.5 0.5  0.5  7.09 3400 0.02 0.03
i 6   19.5 19.5 0.5  0.5  7.06 3400 0.02 0.03
i 2   20   20   0.5  0.5  9.11 3400 0.02 0.03
i 3   20   20   0.5  0.5  9.08 3400 0.02 0.03
i 5   20   20   0.5  0.5  7.08 3400 0.02 0.03
i 6   20   20   0.5  0.5  7.04 3400 0.02 0.03
i 2   20.5 20.5 0.5  0.5  9.09 3400 0.02 0.03
i 3   20.5 20.5 0.5  0.5  9.06 3400 0.02 0.03
```

```
i 5  20.5 20.5  0.5  0.5  7.09 3400 0.02 0.03
i 6  20.5 20.5  1    1    0.00    0 0.02 0.03
i 2  21   21    0.5  0.5  9.08 3400 0.02 0.03
i 3  21   21    0.5  0.5  9.04 3400 0.02 0.03
i 5  21   21    1    1    7.11 3400 0.02 0.03
i 7  21   21    1    1    6.11 3400 0.02 0.03
i 8  21   21    1    1    5.11 3800 0.02 0.03
i 1  21.5 21.5  0.5  0.5  8.09 3400 0.02 0.03
i 2  21.5 21.5  0.5  0.5  9.06 3400 0.02 0.03
i 3  21.5 21.5  0.5  0.5  9.03 3400 0.02 0.03
i 1  22   22    0.5  0.5  8.08 3400 0.02 0.03
i 2  22   22    1.5  1.5  9.04 3400 0.02 0.03
i 4  22   22    1    1    0.00    0 0.02 0.03
i 5  22   22    0.5  0.5  8.04 3400 0.02 0.03
i 7  22   22    0.5  0.5  7.01 3400 0.02 0.03
i 8  22   22    0.5  0.5  6.01 3800 0.02 0.03
i 1  22.5 22.5  0.5  0.5  8.04 3400 0.02 0.03
i 5  22.5 22.5  0.5  0.5  8.03 3400 0.02 0.03
i 7  22.5 22.5  0.5  0.5  6.11 3400 0.02 0.03
i 8  22.5 22.5  0.5  0.5  5.11 3800 0.02 0.03
i 1  23   23    0.5  0.5  8.06 3400 0.02 0.03
i 5  23   23    0.5  0.5  8.01 3400 0.02 0.03
i 7  23   23    0.5  0.5  6.09 3400 0.02 0.03
i 8  23   23    0.5  0.5  5.09 3800 0.02 0.03
i 1  23.5 23.5  0.5  0.5  8.08 3400 0.02 0.03
i 3  23.5 23.5  1    1    9.04 3400 0.02 0.03
i 5  23.5 23.5  0.5  0.5  7.11 3400 0.02 0.03
i 7  23.5 23.5  0.5  0.5  6.08 3400 0.02 0.03
i 8  23.5 23.5  0.5  0.5  5.08 3800 0.02 0.03
i 1  24   24    0.5  0.5  8.09 3400 0.02 0.03
i 2  24   24    0.5  0.5  9.09 3400 0.02 0.03
i 5  24   24    3    3    7.09 3400 0.02 0.03
i 7  24   24    3    3    6.06 3400 0.02 0.03
i 8  24   24    3    3    5.06 3800 0.02 0.03
i 1  24.5 24.5  0.5  0.5  8.11 3400 0.02 0.03
i 2  24.5 24.5  0.5  0.5  9.08 3400 0.02 0.03
i 3  24.5 24.5  0.5  0.5  9.03 3400 0.02 0.03
i 1  25   25    0.5  0.5  8.09 3400 0.02 0.03
i 2  25   25    0.5  0.5  9.06 3400 0.02 0.03
i 3  25   25    1    1    9.01 3400 0.02 0.03
i 1  25.5 25.5  0.5  0.5  8.08 3400 0.02 0.03
i 2  25.5 25.5  0.5  0.5  9.04 3400 0.02 0.03
i 1  26   26    0.5  0.5  8.06 3400 0.02 0.03
i 2  26   26    0.5  0.5  9.03 3400 0.02 0.03
i 3  26   26    1    1    8.09 3400 0.02 0.03
i 1  26.5 26.5  0.5  0.5  8.03 3400 0.02 0.03
i 2  26.5 26.5  0.5  0.5  9.01 3400 0.02 0.03
i 1  27   27    2    2    8.04 3400 0.02 0.03
i 2  27   27    0.5  0.5  8.11 3400 0.02 0.03
i 3  27   27    0.5  0.5  8.08 3400 0.02 0.03
i 5  27   27    0.5  0.5  7.11 3400 0.02 0.03
i 7  27   27    0.5  0.5  6.08 3400 0.02 0.03
i 8  27   27    0.5  0.5  5.08 3800 0.02 0.03
i 2  27.5 27.5  0.5  0.5  9.01 3400 0.02 0.03
i 5  27.5 27.5  0.5  0.5  8.06 3400 0.02 0.03
i 7  27.5 27.5  0.5  0.5  6.09 3400 0.02 0.03
i 8  27.5 27.5  0.5  0.5  5.09 3800 0.02 0.03
i 2  28   28    1    1    8.11 3400 0.02 0.03
i 3  28   28    1    1    8.08 3400 0.02 0.03
i 5  28   28    4    4    7.11 3400 0.02 0.1
i 7  28   28    2    2    6.11 3400 0.02 0.03
i 8  28   28    2    2    5.11 3800 0.02 0.03
i 1  29   29    1    1    8.03 3400 0.02 0.03
i 2  29   29    0.75 0.75 8.09 3400 0.02 0.03
i 3  29   29    1    1    8.06 3400 0.02 0.03
i 2  29.75 29.75 0.25 0.25 8.08 3400 0.02 0.03
i 1  30   30    2    2    8.04 3400 0.02 0.1
i 2  30   30    2    2    8.08 3400 0.02 0.1
i 3  30   30    4.5  4.5  0.00    0 0.02 0.03
i 7  30   30    2    2    6.04 3400 0.02 0.1
i 8  30   30    2    2    5.04 3800 0.02 0.1
i 1  32   32    2.5  2.5  0.00    0 0.02 0.1
i 2  32   32    2.5  2.5  0.00    0 0.02 0.1
i 5  32   32    2.5  2.5  0.00    0 0.02 0.1
i 7  32   32    2.5  2.5  0.00    0 0.02 0.1
i 8  32   32    2.5  2.5  0.00    0 0.02 0.1
f 0  35.5 35.5
e
```

The orchestra file for the Mendelssohn is below. The original file had reverberation; this was removed for making the test piece.

```
;orchestra6
sr=32000
kr=2000
ksmps=16
nchnls=1

instr 1
a1 envlpx p5 ,p6 ,p3 ,p7 ,2 ,1 ,0.01
ga1 oscili a1 ,cpspch(p4) ,1
endin
instr 2
a1 envlpx p5 ,p6 ,p3 ,p7 ,2 ,1 ,0.01
ga2 oscili a1 ,cpspch(p4) ,1
endin
instr 3
a1 envlpx p5 ,p6 ,p3 ,p7 ,2 ,1 ,0.01
ga3 oscili a1 ,cpspch(p4) ,1
endin
instr 4
a1 envlpx p5 ,p6 ,p3 ,p7 ,2 ,1 ,0.01
ga4 oscili a1 ,cpspch(p4) ,1
endin
instr 5
a1 envlpx p5 ,p6 ,p3 ,p7 ,2 ,1 ,0.01
ga5 oscili a1 ,cpspch(p4) ,1
endin
instr 6
a1 envlpx p5 ,p6 ,p3 ,p7 ,2 ,1 ,0.01
ga6 oscili a1 ,cpspch(p4) ,1
endin
instr 7
a1 envlpx p5 ,p6 ,p3 ,p7 ,2 ,1 ,0.01
ga7 oscili a1 ,cpspch(p4) ,1
endin
instr 8
a1 envlpx p5 ,p6 ,p3 ,p7 ,2 ,1 ,0.01
ga8 oscili a1 ,cpspch(p4) ,3
endin
instr 9

out ga1+ga2+ga3+ga4+ga5+ga6+ga7+ga8

endin
```

10.13 Appendix M – The BiMouse three-dimensional controller

Here I outline a simple method of making a three-dimensional controller for the PC. The BiMouse is formed from two standard serial mice, and detects the angle of rotation as well as the x and y axes. (Other alternative controllers exist, all with different benefits and drawbacks.[Sawada, Lopez-Lezcano])

It was originally conceived as a more powerful mouse for the User Interface. However, it also has potential for real-time control of three synthesis parameters. The Theremin is played using two control parameters, frequency and amplitude, so a 'soft Theremin' could be controlled using the x-y position of a mouse. The third dimension available with the BiMouse could control another parameter such as vibrato. In fact, on PCs with all four[91] serial ports, a user could control six continuous dimensions independently (such as pitch, amplitude, attack rate, attack synchrony, and two formant frequencies) with a BiMouse in each hand.[92] To extend this, we could use the mouse buttons to control the mapping from these six dimensions onto many more synthesis parameters, although we would wish to reserve at least one button for specifying note onsets.

10.13.1 BiMouse hardware

The BiMouse is made using two standard and ideally identical mice, as shown in Figure 155. While more elegant attachment is undoubtedly possible, a practical method of joining the mice is to use a thick slice of Blu-Tak. Ideally, the horizontal and vertical axes of the mice should be aligned. The cables of the mice are joined by twist-ties. A more refined system could use optical mice to free the performer from the cables, and a larger mouse mat.

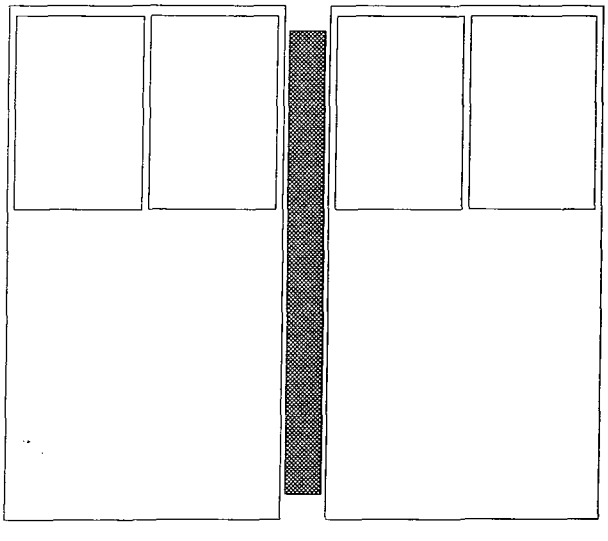


Figure 155 - The hardware of the BiMouse.

10.13.2 BiMouse software

Since the hardware configuration is unusual and a serial port is more

[91] Four serial ports is normally the maximum on a PC – most systems have only two ports. However, an interesting installation by Okamoto at the 1996 ICMC used specially-designed hardware to allow sixteen mice to control real-time polyphonic synthesis.[Okamoto]

[92] Another experimental device along similar lines to the BiMouse was the BiJoy – two joysticks connected at the base. This allows four continuous controllers. It also has four switches, the maximum permitted by the joystick port.

frequently required for a MIDI interface, it is most practical to use an AUTOEXEC.BAT with multiple configurations.

It is necessary for CONFIG.SYS to load two mouse drivers. The Microsoft driver, MOUSE.SYS, checks for itself in memory, so we must circumvent this feature. This is done by copying MOUSE.SYS to MOUSETWO.SYS and loading the latter first. Both drivers should use a linear mapping from mouse movement to cursor position, i.e. dynamic tracking must be disabled.

```
rem load COM1 handler - mousetwo.sys is a direct copy of mouse.sys
DEVICE      = m:\hardware\mouse\mousetwo.sys /1 /R0
rem load COM2 handler
DEVICEHIGH = m:\hardware\mouse\mouse.sys     /2 /R0
```

(Extract from CONFIG.SYS)

10.13.3 Using the BiMouse

User programs must set up their own routines to handle both mouse interrupts. If the mice track at different speeds, then it will be necessary to first compensate for this. The position of the BiMouse is taken from one mouse, and the change in angle is calculated from the difference in Δy between the two mice. The 'fourth' degree of freedom, that we have lost, is the distance between the two balls, which *should* remain constant as long as both are in contact with the mat.

If the left mouse moves by $\Delta x_L, \Delta y_L$, and the right mouse moves by $\Delta x_R, \Delta y_R$, and the BiMouse is at 0position x_B, y_B pointing at an angle a, then:-

$$\Delta x_B = \Delta x_L.\cos(a) + \Delta y_L.\sin(a)$$

$$\Delta y_B = \Delta x_L.\sin(a) - \Delta y_L.\cos(a)$$

$$\Delta a = k.(\Delta y_L - \Delta y_R)$$

A program was written to test this system for two mice. Further developments will be to implement four mice, to develop real-time synthesis routines, and to develop a system to map the gestural control onto synthesis control.

10.14 Appendix N - Colour figures

Below are some of the colour schemes available with READSPEC – 16-colour, 16-grey, blue/yellow, 'fire', and reversed 16-grey. The printed colours only correspond approximately to the screen colours.

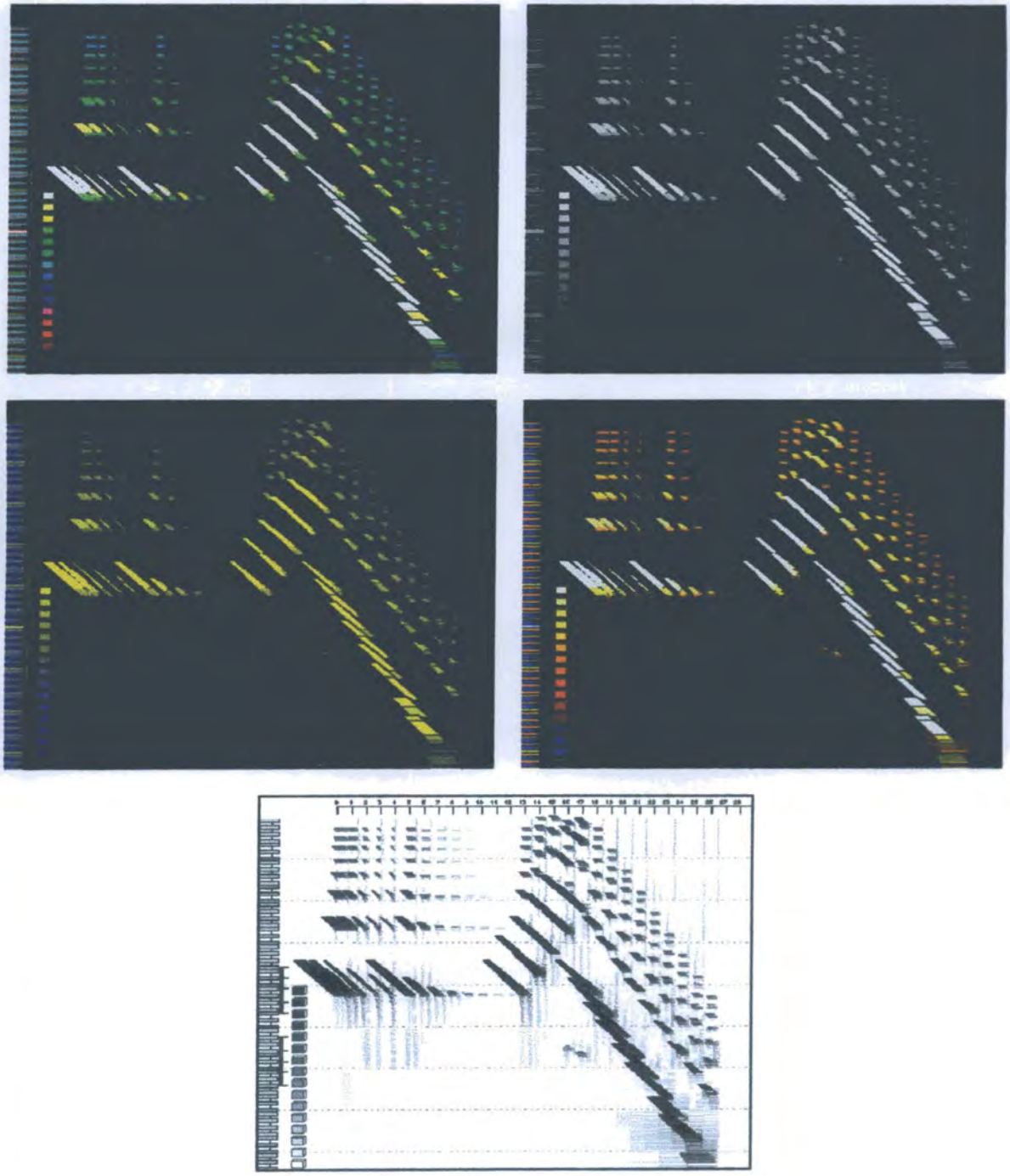


Figure 156 - Colour schemes available with READSPEC.

10.15 Appendix O – Program flags

This appendix gives the usage screens for the main programs developed.

10.15.1 OSA

The C40 program OSA does not take command-line flags; instead it offers the user a menu of options, as shown below. Items without a number depend on other choices and cannot be altered directly by the user.

```
                                           \  \  \  \  \  \  \  \  \  \  \  \  \
            Octave Spectral Analysis        \  \_\  \_\  \  \_\  \_\  \_\  \
                                             \_    \_    \_\_    \_    \_    \_\_

 Current configuration is as follows:-

  1> Size of disk buffer (words)         4096
  2> Size of channel buffer (words)      128
  3> Filter number                       6
   - Length of filter                    128
  5> Length of FFT                       64
  6> Input file                          i:\mex\poulenc\poulenc.snd
  7> Input file centred on 0             yes
   - Input file length (seconds)         123456
  9> Calculate spectra                   yes
 10> Write spectra                       yes
 11> Spectrum format                     IEEE
 12> Output file                         i:\mex\poulenc\poulenc.c40
 13> Sample rate (Hz)                    16000
 14> FFT enabled                         yes
 15> Window type                         Hamming
 16> *.WAV format                        no
   - Configuration version               0.08
 18> Task (0=osa, 1=extended)            0
 19> Graphics enabled                    no
 20> Graphics mode                       0
   - X resolution                        0
   - Y resolution                        0
```

10.15.2 ReadSpectrum

```
      ____          ____
     /  __)        /  __)
    / /   __  __  _ ( (__   _  __  __  __  _        __
   / / /_  /_/ /_) \__  \  /_) /_  /    /  /_) / / /7/
  (_/ (_  / / /_/ (____/ /    (_  (_   /  /  \ (_/ / /

(c) Douglas Nunn
Program Version 0.35   Oct 23 1995

This program displays the spectra created by the C40.
To abort early, press escape. Press a key after the beep.

Usage: ReadSp specfile[.c40] [flags]

Flags          (not case-sensitive)
__          __
   -h          Horizontal
__ -t       __ show Time axis
   -c#         Colours 0:8-col 1:mono 2:8-gy [3:16-col] 4:16-gy 5:b/y 6:fire
   -m          use Mono colour scheme (same as -c1)
   -r          Reverse colours
   -k          Kolour by semitones
   -s#         Style number (0-12, default 10)
__ -sl#     __ SLant angle (0-90, default 40.00)
```

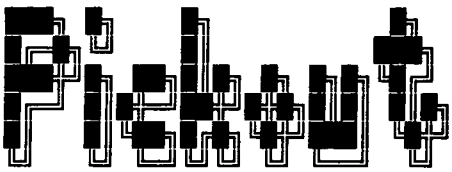
```
    -p          Photo mode - pause after each scroll (for VGACAP)
    -pp         Photo Pcx - use PCXDUMP to capture screen
    -ps         Photo Shell (with -p or -pp) - run RS_Shell.Bat after snap
    -sc#        SCroll every n pixels (default 16) (only with -h)
    -qx         Quick eXit - run once, don't pause at end
__  -n      __  No scrolling
__  -l      __  show Loudness instead of amplitude (?)
    -ename      write Edge file
__  -d      __  Draw edges
__  -?      __  this help
```

10.15.3 Characterisation – Distrib2.Exe

```
Syntax: Distrib2.Exe SpectrumFile [OutputFile]
This graphically displays the power distribution and determines the
average power.

SpectrumFile is a *.c40 file.
OutputFile is an optional file for the resultant power calculation.
```

10.15.4 Pickout

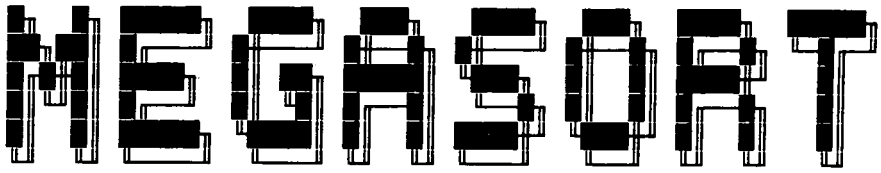
Options marked with an asterisk are the 'normal' options.



```
Version
 0.24
```

```
Syntax: PickOut specfile [flags]

  specfile      spectra output from C40

 Flags:                                      Default
   -?           print this help
*  -m int       Maximum number of tries        40
*  -s float     Sample rate in Hz       44100.000000
*  -g           display Graphics
   -G path      path to graphics driver
*  -t fname     write Text file     (large!)
   -p fname     write Packed file   (useless)
   -b           write Blurb file
   -w           write Waffle file
   -o int       read 1 in n spectra (debug)     1
   -l           Loud - report during run
*  -n int       size of fft                    64
   -q float     Quietness of original in dB     0
*  -k fname     Read character from file *.kar
*  -z float     Target power drop in dB        48
*  -a           Automatic - no user prompts

Example: PickOut bach.c40 -tbach.sla -kcharactr.mex -z24 -m20
Don't use both -q and -k. -p is pointless. Don't use -j either.
```

10.15.5 Megasort



```
                                Version 0.05
Usage: MegaSort input-file output-file [flags]
Flags: -d Disqualify lines not starting with a number
```

10.15.6 Virtual Memory

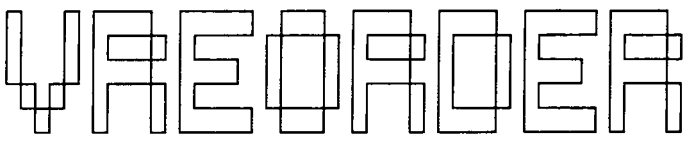
Below is the header file for the virtual memory routines.

```
/* vmem.h --- definitions for Virtual MEMory */

#ifndef VMEM_H
#define VMEM_H

typedef unsigned long VMEM;
#define ulong unsigned long
#define uint unsigned int
#define NULL_VMEM   (ulong)(-1)
VMEM vmalloc(ulong nbytes);
void vfree(VMEM ap);
void vwrite(void *ph, VMEM st, ulong off, ulong n);
void vread(void *ph, VMEM st, ulong off, ulong n);
void vcopy(VMEM src, VMEM dest, ulong dest_off, ulong n);
void vmdie(void);
void vmremove(void);
void vmhush(void);
void vmloud(void);
void vmforce(int neer,int farr,int zz,int extend,int zzz,int disk);
void disallow_overrun(void);
void allow_overrun(void);

extern char VMdlist[33];
extern int VMdlistn;

#endif
```

10.15.7 Sine Display

This is a QBasic program, and QBasic programs cannot be given flags. If flags must be passed, they must be read from a particular file. Instead, this program assumes that the input file is found in D:\MEX\MEX.SLB.

10.15.8 Vreorder



```
Version 0.07
```

```
VReorder sorts a sine list according to a chain list.

Usage: VReorder list-file chain-file output-file
     list-file is a sorted list of sinusoids     (*.slb)
    chain-file is the list of chain connections  (*.chn)
   output-file is the reordered output           (*.reo)
```

10.15.9 Track display

This is also a QBasic program, so has no flags.

10.15.10 Sine Tracking

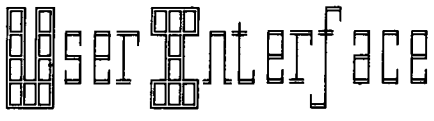
```
 ### ##  ### ##  # # ###   #  ## # #     ### ##   #   ## # # ### ##
 ##  ##  ##  ##  # # ##  ### #    #       #  ##  ### #   ##  ##  ##
 #   # # ###   # ### ### #    ## #        #  # # # #  ## # # ### # #


Program version 1.05
FT.Exe matches harmonically related sines into note groups.

Usage: FT input-file [[dumpfilename] [-dt#]
      input-file   (*.reo)          - Grouped sine format
      dumpfilename (NO extension!) - name for vmem dump files
      -dt#         (#=16,32,64)    - DivideTime factor (=FFTsize)
      -nk                          - No Keypresses or prompts
```

10.15.11 Battle


```
Version
 0.28

Syntax: Batl TempDumpName AsciiFile [flags]

 TempDumpName (with NO extension) refers to .own/.clm/.seg files.
 AsciiFile is the ASCII note list produced.

 Flags:-
   -snumber    Sample rate
   -q          Quiet - no other screen info
   -tnumber    Minimum time for a note in seconds
   -b          Bleep at end of each generation
   -nk         No Keypresses (auto mode for batch files)
   -rfname     Report File
```

10.15.12 User Interface

Most of the UI parameters are controlled via the menu system, but there are several startup flags.




```
UI - User Interface version 0.26    (c) Douglas Nunn, 4 August 1996

Usage: UI [flags]
```

```
-m1       Mouse on COM1
-m2       Mouse on COM2
-gd#      Graphics Driver (-1:VGA 0:SVGA16 1:SVGA256 5:S3 6:Twk16 7:Twk256)
-gm#      Graphics Mode
-gpPATH   Path to graphics drivers
-ps       Pause at Start
-?        This help

          Default mouse port is COM2.
```

10.15.13 ReadAsc

```
ReadAsc version 0.02


ReadAsc plots Cakewalk ASC files.

Syntax:  ReadAsc [options] [flags] infile1 [flags] [infile2 ... ]

Options: -cp   - ComPare infile1 (guessed) to infile2 (ideal)
         -sh   - SHow comparison graphically

Flags:   -c#   - Colour
         -bg#  - BackGround colour
         -xs#  - X Scale      default 5.000000
         -nw#  - Note Width   default 5.000000
         -ln#  - Lowest Note  default 20

0-black   1-blue   2-green   3-cyan   4-red   5-magenta   6-brown   7-Lgrey
8-darkgrey 9-Lblue 10-Lgreen 11-Lcyan 12-Lred 13-Lmagenta 14-yellow 15-white

Example: readasc -c1 origscor.asc -c4 newscor.asc
```

10.16 Appendix P - Converting the sinc function to Gabor wavelets

Here I derive sinc(x) in terms of Gabor wavelets.

First we try to form $Aim(x) := \frac{1}{\pi \cdot x}$

If we sum terms from negative to positive $minJ := 20$ $maxJ := 29$

so the number of terms is $J := minJ + maxJ + 1$ $Q := J$

and the index range is $j := -minJ .. maxJ$

we also sum over positive and negative times $q := 0 .. 1$ $sign(q) := 1 - 2 \cdot q$

The ratio for spacing is $R := 1.371$

We form quanta at times $t_{j + minJ + q \cdot Q} := R^{j} \cdot sign(q)$

with a frequency of zero $f := 0$

with densities $a_{j + minJ + q \cdot Q} := \left(R^{2}\right)^{-j}$

and magnitudes $m_{j + minJ + q \cdot Q} := R^{-j} \cdot sign(q)$

The last term is numbered $lastJ := J \cdot 2 - 1$ $lastJ = 99$

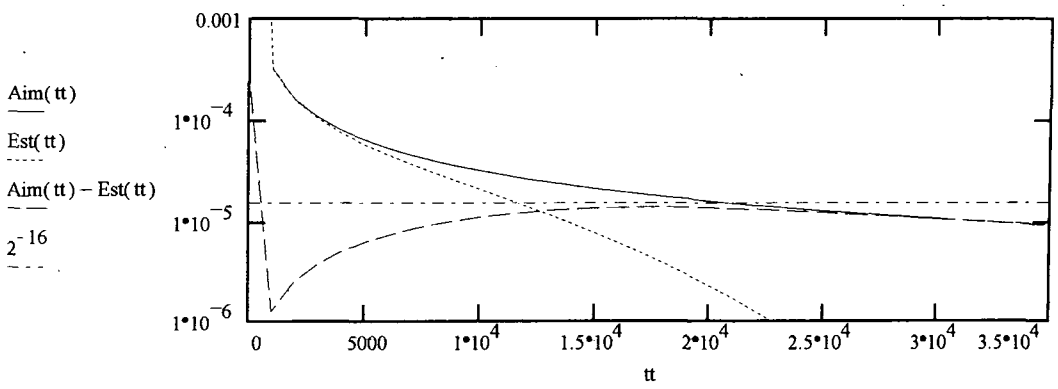
and the total of these terms summed over $jj := 0 .. lastJ$

is given by $est0(tt) := \sum_{jj=0}^{lastJ} m_{jj} \cdot e^{-a_{jj} \cdot (tt - t_{jj})^2}$

Now determine the constant factor $Kappa := \frac{Aim(10.7)}{est0(10.7)}$ $Kappa = 0.067244$

and apply it to give the final estimate $Est(tt) := est0(tt) \cdot Kappa$

plot for time index $tt := 0.01, 1000 .. 35000$

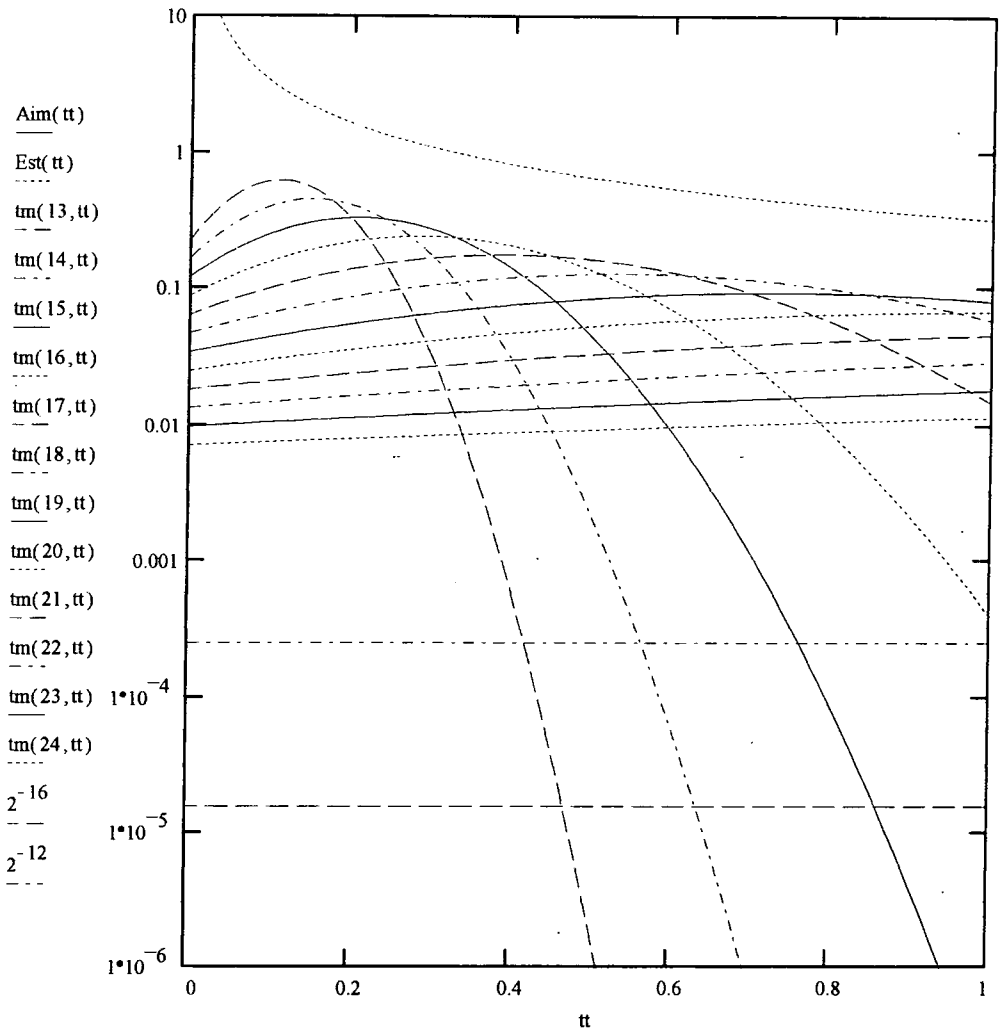


This shows that the absolute error does not exceed 2^{-16} when we truncate the end of the series

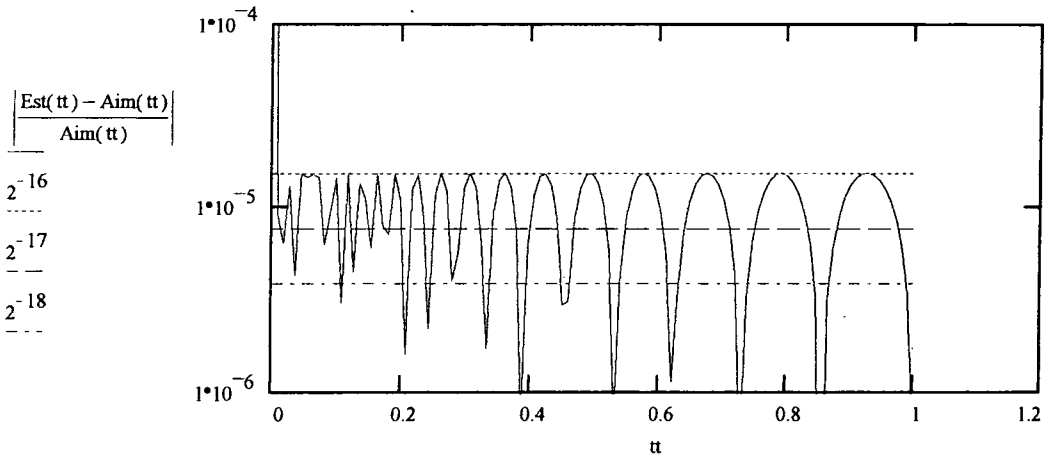
The individual terms are $tm(x, tt) := m_x \cdot e^{-a_x \cdot (tt - t_x)^2} \cdot Kappa + 2^{-99}$

Plot this for times

$tt := 0.001, .01 .. 1$



The relative error is also below 2^{-16}



10.17 Appendix Q – Audio examples

These audio examples can be found on the enclosed cassette tape and on the World-Wide Web at the URL http://capella.dur.ac.uk/doug/thesis/.

	Name		*File*	*Threshold (dB)*
1	MTest1	original	d:\mex\mtest1\mtest1	
2	MTest2	original	d:\mex\mtest2\mtest2.snd	
3	"	derived	d:\mex\mtest2\mex.wrk	–24
4	Mendel	original	d:\mex\minimend\minimend.snd	
5	"	derived	d:\mex\mendhamm\mex\redo_96\batl30.wrk	–30
6	Poulenc	original	d:\mex\poulenc\poulenc.snd	
7	Träumerei	original	d:\mex\schum\schum.snd	
8	"	derived	d:\mex\schum\redo96\batl42.wrk	–42
9	Piano Concerto	original	d:\mex\gpc\gpc.snd	
10	"	derived	d:\mex\gpc\batl18.wrk	–18
11	Death of Aase	original	d:\mex\aase\aase.snd	
12	"	derived	d:\mex\aase\batl24.wrk	–24
13	Didgeridoo	original	d:\mex\didge\didge.snd	
14	Ringdown	original	d:\mex\bells3\bellmon.snd	

Table 48 - List of audio examples.

11. References

11.1 References

Amuedo — **J. Amuedo,** *Periodicity estimation by hypothesis-directed search,* Proc. ICASSP, Tampa, Florida, 395-398, 1985

Analog — **Analog Devices,** *ADSP-21020 User's Manual,* 1991

ANSI — **American National Standards Institute,** *American Standard Acoustical Terminology,* S1.1-1960, 1960

Arfib 90 — **D. Arfib,** *In the intimacy of a sound,* Proc. ICMC, Glasgow, 43-45, 1990

Arfib 91 — **D. Arfib,** *Analysis, transformation, and resynthesis of musical sounds with the help of a time-frequency representation,* in G. De Poli, A. Piccialli, and C. Roads, eds., Representations of Musical Signals, Cambridge, Massachusetts: MIT Press, 1991

Armani — **F. Armani, A. Paladin, and C. Rosati,** *MARS Applications Using APPL120 Development Tools: a Case Study,* Proc. ICMC, Aarhus, 230-236, 1994

Assmann 89 — **P.F. Assmann and Q. Summerfield,** *Modeling the perception of concurrent vowels: Vowels with the same fundamental frequency,* JASA, 85, 327-338, 1989

Assmann 90 — **P.F. Assmann and Q. Summerfield,** *Modeling the perception of concurrent vowels: Vowels with different fundamental frequencies,* JASA, 88, 680-697, 1990

Athena — **Athena Consulting,** *Athena* Consulting Stock Answers: How FAST are Athena workstations?, <http:// mufasa.mit.edu/ stock_answers/ workstations/ ws_speeds.html>, 27 September 1996

Atlanta — **Atlanta Signal Processors Inc.,** *Digital Filter Design Package,* <http://www.aspi.com/techsprt/dfdp.htm>, November 1996

AudioWorks — **AudioWorks Ltd.,** *Sound2Midi Info (sales literature),* <http:// www.audioworks.com/s2m.htm>, 23 June 96

Auger — **F. Auger and G. Flandrin,** *La réallocation: une méthode générale d'amélioration de la lisibilité des représentation temps-fréquence bilinéaires,* Proc. Colloque Temps-Fréquence, Ondelettes et Multiresolution: Théorie, Modeles et Applications, Lyon, France, 15.1-15.7, 1994 [in French]

Backus — **J. Backus,** *Input impedance curves for the brass instruments,* JASA, 60(2), 470-480, August 1976

Bailey 90 — **N. Bailey, A. Purvis, I. Bowler, and P.D. Manning,** *An highly parallel architecture for real-time music synthesis and digital signal processing applications,* Proc. ICMC, Glasgow, 167-171, 1990

Bailey 91 — **N. Bailey,** *On the synthesis and processing of high-quality audio signals by parallel computers,* PhD thesis, Durham Univ., 1991

Baines — **A. Baines,** *Brass Instruments,* London: Faber & Faber, 1976

Baird — **B. Baird, D. Blevins, and N. Zahler**, *Artificial intelligence and music: implementing an interactive computer performer*, CMJ, 17(2), 73-79, Summer 1993

Balzano — **G.J. Balzano**, *The Group-theoretic Description of 12-Fold and Microtonal Pitch Systems*, CMJ, 4(4), 66-84, Winter 1980

Baraniuk — **R.G. Baraniuk and D.L. Jones**, *Shear Madness: new orthonormal bases and frames using chirp functions*, IEEE Trans. Signal Processing, 41(12), 3543-3548, December 1993, and <http://www-dsp.rice.edu/publications/pub/shear_madness.ps.Z>

Bargar 92 — **R. Bargar,** *Correlated sound and image in a digital medium*, Proc. ICMC, San Jose, 194-197, 1992

Bargar 95 — **R. Bargar, B. Holloway, X. Rodet, and C. Hartman**, *Defining Spectral Surfaces*, Proc. ICMC, Banff, 373-376, 1995

Bartoo — **T. Bartoo and B. Truax**, *Electro-Acoustic Composer's Workstation Project*, Proc. ICMC, San Jose, 446, 1992

Bastiaans 80 — **M.J. Bastiaans**, *Gabor's Expansion of a Signal into Gaussian Elementary Signals*, Proc. IEEE, 68(4), 538-539, April 1980

Bastiaans 85 — **M.J. Bastiaans**, *On the sliding-window representation in digital signal processing*, IEEE Trans. ASSP, 33(4), 1985

Beauchamp — **J. Beauchamp**, *A Computer System for Time-Variant Harmonic Analysis and Synthesis of Musical Tones*, in Music By Computers, H. von Foerster and J. Beauchamp, eds., New York: John Wiley & Sons, 1969

Bell — **A.J. Bell and T.J Sejnowski**, *An information-maximisation approach to blind separation and blind deconvolution*, Neural Computation, 7, 1129-1159, 1995, and <ftp://ftp.cnl.salk.edu/pub/tony/bell.blind.ps.Z>

Bellanger — **M. Bellanger, J. Daguet, and G. Lepagnol**, *Interpolation extrapolation and reduction of computation speed in digital filters*, IEEE Trans. ASSP, 22, 231-235, August 1974

Benade 65 — **A.H. Benade and J.W. French**, *Analysis of the Flute Head Joint*, JASA, 37, 679-691, 1965

Benade 73 — **A.H. Benade**, *The Physics of Brasses*, Scientific American, 229(1), 24-35, July 1973, reprinted in Musical Acoustics – Piano and Wind Instruments, E.L. Kent, ed., Stroudsburg, Pennsylvania: Dowden, Hutchinson and Ross, 1977

BergerJ 94a — **J. Berger, R.R. Coifman, and M.J. Goldberg**, *A Method of Denoising and Reconstructing Audio Signals*, Proc. ICMC, Aarhus, 344-347, 1994

BergerJ 94b — **J. Berger and C. Nichols**, *Using Wavelet Based Analysis and Resynthesis to Uncover the Past*, Proc. ICMC, Aarhus, 352-355, 1994

BergerJ 94c — **J. Berger, R.R. Coifman, and M.J. Goldberg,** *Removing Noise from Music Using Local Trigonometric Bases and Wavelet Packets*, J. AES, 42(10), 808-818, 1994

BergerJ 95 — **J. Berger, R.R. Coifman, and M.J. Goldberg,** *A two-stage automatic adaptive process to remove noise from an audio signal*, Proc. ICMC, Banff, 288-291, 1995, and <http://www.music.yale.edu/research/denoise/denoise2.html>

BergerK — **K.W. Berger,** *Some Factors in the Recognition of Timbre*, JASA, 36, 1888, 1963

Bevan — **C. Bevan,** *The Tuba Family*, London: Faber & Faber, 1978

Birdsall — **J.W. Birdsall** <support@picarefy.com>, *XMSIF*, <ftp://sunsite.doc.ic.ac.uk/packages/simtelnet/msdos/c/xmsif15.zip>, 1994

Blackham — **E.D. Blackham,** *The Physics of the Piano*, in Musical Acoustics – Piano and Wind Instruments, E.L. Kent, ed., Stroudsburg, Pennsylvania: Dowden, Hutchinson and Ross, 1977

Bloch — **J.J. Bloch and R.B. Dannenberg,** *Real-Time Computer Accompaniment of Keyboard Performances*, Proc. ICMC, ?, 279-290, 1985

Boatin — **N. Boatin, G. De Poli, and P. Prandoni**, *Timbre Characterization with Mel-Cepstrum: a Multivariate Analysis,* Proc. XI Colloquium on Musical Informatics, Bologna, 145-148, 1995

Borland — **Borland Inc.,** *Turbo C Reference Guide*, Scotts Valley: Borland, 1988

Bosi — **M. Bosi,** *The Sound Accelerator as a Real-Time DSP Environment: Encoding/Decoding Audio Signals*, Proc. ICMC, Glasgow, 175-177, 1990

Boyer — **F. Boyer and R. Kronland-Martinet,** *Granular Resynthesis and Transformation of Sounds Through Wavelet Transform Analysis,* Proc. ICMC, Ohio State U., 51-54, 1989

Bracewell — **R.N. Bracewell**, *The Fourier Transform and its Applications*, New York: McGraw-Hill, 1986

Bregman 89 — **A.S. Bregman,** *Auditory Scene Analysis: The Perceptual Organization of Sound,* London: MIT Press, 1989

Bregman 96a — **A.S. Bregman,** *Psychological data and computational ASA,* in H. Okuno and D.F. Rosenthal, eds., Readings in Computational Auditory Scene Analysis, Hillsdale: Erlbaum, 1996

Bregman 96b — **A.S. Bregman,** *Introduction to Auditory Scene Analysis,* <http://www.psych.mcgill.ca/labs/auditory/introASA.html>, April 1996

BrownG 92a — **G.J. Brown**, *Computational Auditory Scene Analysis: A Representational Approach,* PhD thesis CS-92-22, Dept. of Comp. Sci., Univ. of Sheffield, 1992

BrownG 92b — **G.J. Brown and M.P. Cooke**, *Computational Auditory Scene Analysis: Grouping sound sources using common pitch contours,* Proc. Inst. Acoustics, Windermere, 439-446, November 1992

BrownG 94a — **G.J. Brown and M.P. Cooke**, *Perceptual Grouping of Musical Sounds,* JNMR, 23 (2), 107-132, 1994

BrownG 94b — **G.J. Brown and M.P. Cooke**, *Computational Auditory Scene Analysis,* Computer Speech and Language, 8, 297-336, 1994

BrownG 95a — **G.J. Brown and M.P. Cooke**, *A neural oscillator model of primitive auditory grouping,* Proc. IEEE Sig. Proc. Society Workshop on Applications of Sig. Proc. to Audio and Acoustics, New York, 53-56, October 1995

BrownG 95b — **G.J. Brown and M.P. Cooke**, *Temporal synchronisation in a neural oscillator model of primitive auditory stream segregation,* IJCAI Workshop on Computational Auditory Scene Analysis, Montreal, August 1995

BrownG 96 — **G.J. Brown, M.P. Cooke, and E. Mousset,** *Are neural oscillations the substrate of auditory grouping?*, ESCA Tutorial and Workshop on the Auditory Basis of Speech Perception, Keele, July 1996, and <ftp://ftp.dcs.shef.ac.uk/ share/spandh/pubs/brown/bcm-keele96.ps.Z>

BrownJ 87 **J.C. Brown and M.S. Puckette,** *Musical Information from a Narrowed Autocorrelation Function,* Proc. ICMC, Urbana, 84-88, 1987

BrownJ 89 **J.C. Brown and M.S. Puckette,** *Calculation of a "narrowed" autocorrelation function,* JASA, 85(4), 1595-1601, 1989

BrownJ 91a **J.C. Brown,** *Calculation of a Constant Q Spectral Transform,* JASA, 89, 425-434, 1991

BrownJ 91b **J.C. Brown and B. Zhang,** *Musical pitch tracking using the methods of conventional and "narrowed" autocorrelation,* JASA, 89(5), 2346-2354, 1991, and <http://sound.media.mit.edu/~brown/try/try.html>

BrownJ 92 **J.C. Brown,** *Musical fundamental frequency tracking using a pattern recognition method,* JASA, 92(3), 1394-1402, 1992

Bürck **W. Bürck, P. Kotowski, and H. Lichte,** *Die Hörbarkeit von Laufzeitdifferenzen,* Elektrotechn. Nachr.-Techn., 12, 355, 1935 [in German]

Burns **E.M. Burns and W.D. Ward,** *Intervals, scales, and tuning,* in The Psychology of Music, D. Deutsch, ed., 241-269, New York: Academic Press, 1982

Butler **D. Butler,** *A further study of melodic channeling,* Perception and Psychophysics, 25, 264-268, 1979

Calway **A. Calway,** *The Multiresolution Fourier Transform: A general Purpose Tool for Image Analysis,* PhD thesis, Dept. of Comp. Sci., Univ. of Warwick, UK, September 1989

Carver **N. Carver and V. Lesser,** *Blackboard systems for knowledge-based signal understanding,* in A. Oppenheim and S. Nawab, eds., Symbolic and Knowledge-based Signal Processing, London: Prentice Hall, 1992

Chafe 82 **C. Chafe, B. Mont-Reynaud, and L. Rush,** *Toward an intelligent editor of digital audio: Recognition of musical constructs,* CMJ, 6(1), 30-41, Spring 1982

Chafe 85 **C. Chafe, D. Jaffe, K. Kashima, B. Mont-Reynaud, and J. Smith,** *Techniques for Note Identification in Polyphonic Music,* Proc. ICMC, Vancouver, 399-405, 1985, and Stanford Music Dept. Technical Report STAN-M-29

Chafe 86 **C. Chafe and D. Jaffe,** *Source Separation and Note Identification in Polyphonic Music,* Proc. ICASSP, vol. 2, 1289-1292, Tokyo, 1986

Chafe 91 **C. Chafe,** *Simulating Performance on a Bowed Instrument,* in Current Directions in Computer Music Research, M.V. Mathews and J.R. Pierce, eds., 185-198, Cambridge, Massachusetts: MIT Press, 1991

Chan **D.C.B. Chan, P.J.W. Rayner, and S.J. Godsill,** *Multi-channel Blind Source Separation By Decorrelation,* IEEE WASPAA, Mohonk, 1995

Chapman **D. Chapman, M. Clarke, M. Smith, and P. Archbold,** *Self-Similar Grain Distribution: A Fractal Approach to Granular Synthesis,* Proc. ICMC, Hong Kong, 212-213, 1996

ChenM **M. Chen** <mchen@cse.psu.edu>, *Gus Tester,* <ftp://ftp.cdrom.com/pub/gus/util/dos/gustest.zip>, April 1993

ChenS 94 **S.S. Chen,** *Basis Pursuit*, Proc. 28th Asilomar Conference on Signals, Systems, and Computers, vol. 1, 41-44, 1994, and <http://playfair.stanford.edu/reports/chen_s/asilomar.ps.Z>

ChenS 95 **S.S. Chen,** *Basis Pursuit*, PhD thesis, Dept. of Statistics, Stanford Univ., November 1995, and <http://playfair.stanford.edu/reports/chen_s/thesis.ps.Z>

ChenS 96 **S.S. Chen, D.L. Donoho, and M.A. Saunders,** *Atomic Decomposition by Basis Pursuit*, <http://playfair.stanford.edu/reports/chen_s/BasisPursuit.ps.Z> and Stanford Statistics Dept. Technical Report, 1996

Childers **D.G. Childers and C.K. Lee,** *Co-Channel Speech Separation*, Proc. ICASSP, 6.4.1-6.4.4, 1987

Choi **A. Choi,** *A Least-Square Algorithm for Fundamental Frequency Estimation*, Proc. ICMC, Banff, 284-287, 1995

Chowning 73 **J.M. Chowning,** *The Synthesis of Complex Audio Spectra by Means of Frequency Modulation*, J. AES, 21(7), 526-534, 1973 (reprinted in CMJ, 1(2), 46-54, Summer 1977)

Chowning 84 **J.M. Chowning, L. Rush, B. Mont-Reynaud, C. Chafe, A. Schloss, and J.O. Smith,** *Intelligent Systems for the Analysis of Digitized Acoustic Signals, Final Report*, Stanford Music Dept. Technical Report STAN-M-15, 1984

Chowning 86a **J.M. Chowning and D. Bristow,** *FM Theory and Applications*, Yamaha Music Foundation, Tokyo, 1986

Chowning 86b **J.M. Chowning and B. Mont-Reynaud,** *Intelligent Analysis of Composite Acoustic Signals*, Stanford Music Dept. Technical Report STAN-M-36, 1986

Chowning 93 **J.M. Chowning,** *Computer Music: A Grand Adventure and Some Thoughts About Loudness*, Proc. ICMC, Tokyo, 2-8, 1993

Chu **P.L. Chu,** *Quadrature Mirror Filter Design for an Arbitrary Number of Equal Bandwidth Channels*, IEEE Trans. ASSP, 33(1), 203-218, February 1985

Churchland **P. Churchland, V.S. Ramachandran, and T. Sejnowski,** *A Critique of Pure Vision*, in Large-Scale Theories of the Brain, C. Koch and J. Davis, eds., Cambridge, Massachusetts: MIT Press, 1994

Cohen **M.M. Cohen and D. Massaro,** *Synthesis of visible speech*, Behaviour Research Methods, Instruments, and Computers, 22(2), 260-263, April 1990

Coifman 90 **R.R. Coifman, Y. Meyer, S. Quake, and M.V. Wickerhauser,** *Signal Processing and Compression with wave packets,* Technical Report, Yale University, Maths Dept., April 1990

Coifman 92 **R.R. Coifman and M.V. Wickerhauser,** *Entropy-Based Algorithms for Best Basis Selection,* Proc. Int. Conf. on Wavelets and Applications, Toulouse, 1992

Cook 88 **P.R. Cook,** *Implementation of Single Reed Instruments with Arbitrary Bore Shapes Using Digital Waveguide Filters*, Stanford Music Dept. Technical Report STAN-M-50, 1988

Cook 91 **P.R. Cook,** *Identification and Control of Parameters in an Articulatory Vocal Tract Model, With Applications to the Synthesis of Singing*, PhD thesis, Dept. of Elec. Eng., Stanford Univ., 1991

Cook 92 **P.R. Cook, D. Morrill, and J.O. Smith,** *An Automatic Pitch Detection and MIDI Control System for Brass Instruments*, ASA conference, New Orleans,

November 1992, and <ftp://ccrma-ftp.stanford.edu/pub/Publications/MIDITrumpetPaper.ps.Z>

Cook 93 — **P.R. Cook, D. Morrill, and J.O. Smith,** *A MIDI Control and Performance System for Brass Instruments*, Proc. ICMC, Tokyo, 130-133, 1993

Cook 96 — **P.R. Cook,** *Physically Informed Sonic Modeling (PhISM): Percussive Synthesis,* Proc. ICMC, Hong Kong, 228-231, 1996

Cooke 91 — **M.P. Cooke,** *Modelling auditory processing and organisation*, PhD thesis, Dept. of Comp. Sci., Univ. of Sheffield, 1991

Cooke 93a — **M.P. Cooke and G. Brown,** *Computational auditory scene analysis: Exploiting principles of perceived continuity*, Speech Communication, 13, 391-399, 1993

Cooke 93b — **M.P. Cooke,** *Modelling auditory processing and organisation*, Cambridge: Cambridge University Press, 1993

Cooke 96a — **M.P. Cooke,** *Auditory Organisation and Speech Perception: Arguments for an Integrated Computational Theory*, ESCA Tutorial and Workshop on the Auditory Basis of Speech Perception, Keele, July 1996, and <http://www.dcs.shef.ac.uk/ research/ groups/ spandh/ martin/ Keele96/KeelePaper96.html>

Cooke 96b — **M.P. Cooke, A. Morris, and P.D. Green,** *Recognising occluded speech*, ESCA Tutorial and Workshop on the Auditory Basis of Speech Perception, Keele, July 1996, and <ftp:// ftp.dcs.shef.ac.uk/ share/ spandh/ pubs/ cooke/KeeleROOS96.ps.Z>

Cover — **T.M. Cover and R.C. King**, *A convergent gambling estimate of the entropy of English*, IEEE Trans. on Information Theory, IT-24(4), 413-421, July 1978

Cox — **M.J. Cox** <mjhc8@eeng.bradford.ac.uk>, *Resplay*, <http://www.ukweb.com/~mark/software.html#ResPlay>, May 1996

Craig — **C. Craig**, *GoldWave (documentation)*, <http://www.goldwave.com/>, May 1997

Crandall — **R. Crandall and M. Minnick**, *Parallel Transform Method for Lossless Compression of Analog Data*, Proc. ICMC, Montreal, 525-528, 1991

Crawford — **M.D. Crawford, M.P. Cooke, and G. Brown,** *Interactive computational auditory scene analysis: An environment for exploring auditory representations and groups*, JASA, 93, 2308, 1993

Culloch — **A.D. Culloch**, *Porting the 3L Parallel C environment to the Texas Instruments TMS320C40*, Transputer Research and Applications 5, in Transputer Research and Applications, A. Veronis and Y. Paker, eds., Amsterdam: IOS Press, 1992

Dannenberg 84 — **R.B. Dannenberg,** *An on-line algorithm for real-time accompaniment,* Proc. ICMC, Paris, 193-198, 1984

Dannenberg 87 — **R.B. Dannenberg and B. Mont-Reynaud**, *Following an Improvisation in Real-Time*, Proc. ICMC, Illinois, 241-248, 1987

Dannenberg 88 — **R.B. Dannenberg and H. Mukaino**, *New Techniques for Enhanced Quality of Computer Accompaniment*, Proc. ICMC, Cologne, 243-249, 1988

Dannenberg 91 — **R.B. Dannenberg**, *Real-time Scheduling and Computer Accompaniment*, in Current Directions in Computer Music Research, M.V. Mathews and J.R. Pierce, eds., 225-261, Cambridge, Massachusetts: MIT Press, 1991

Dannenberg 92 — **R.B. Dannenberg and C.W. Mercer**, *Real-time Software Synthesis on Superscalar Architectures*, Proc. ICMC, San Jose, 174-177, 1992

Dannenberg 93a — **R.B. Dannenberg**, *Music representation issues, techniques, and systems*, CMJ, 17(3), 20-30, Fall 1993

Dannenberg 93b — **R.B. Dannenberg**, *The Implementation of Nyquist, A Sound Synthesis Language*, Proc. ICMC, Tokyo, 168-171, 1993

Darton — **L. Darton,** <darton@macon.wb.slb.com>, *"how to detect COMPRESSED drive?"* <news:comp.sys.ibm.programmer and private email>, 31 October 1995

Daubechies 88a — **I. Daubechies**, *Orthonormal bases of compactly supported wavelets*, Communications in Pure and Applied Mathematics, 41, 1988

Daubechies 88b — **I. Daubechies**, *Time-frequency localization operators: a geometric phase space approach*, IEEE Trans. on Information Theory, 34, 605-612, 1988

Daubechies 90 — **I. Daubechies**, *The Wavelet Transform, time-frequency localization and signal analysis*, IEEE Trans. Information Theory, 36, 961-1005, 1990

Daubechies 92 — **I. Daubechies**, *Ten Lectures on Wavelets*, SIAM, Philadelphia, 1992

Davidson — **G. Davidson, L. Fielder, and M. Antill**, *High Quality Transform Coding at 128 kbits/s*, Proc. ICASSP, Albuquerque, 1990

de Cheveigné 93 — **A. de Cheveigné,** *Separation of concurrent harmonic sounds: Fundamental frequency estimation and a time-domain cancellation model of auditory processing*, JASA, 93, 3271-3290, 1993

de Cheveigné 95 — **A. de Cheveigné, S. McAdams, J. Laroche, and M. Rosenborg,** *Identification of concurrent harmonic and inharmonic vowels: A test of the theory of harmonic cancellation and enhancement*, JASA, 97, 3736-3749, 1995

de Cheveigné 96 — **A. de Cheveigné,** *A neural cancellation model of F_0-guided sound separation*, Proc. ESCA workshop on the Auditory Basis of Speech Perception, Keele, 1996, and <ftp://ftp.linguist.jussieu.fr/people/alain/papers/keele.ps.Z>

Degazio — **B. Degazio**, *Towards a Chaotic Musical Instrument*, Proc. ICMC, Tokyo, 393-395, 1993

Delprat 90 — **N. Delprat, Ph. Guillemain, and R. Kronland-Martinet**, *Parameters estimation for non-linear resynthesis methods with the help of a time-frequency analysis of natural sounds*, Proc. ICMC, Glasgow, 88-90, 1990

Delprat 92 — **N. Delprat, B. Escudié, Ph. Guillemain, R. Kronland-Martinet, P. Tchamitchian, and B. Torrésani**, *Asymptotic Wavelet and Gabor Analysis: Extraction of Instantaneous Frequencies*, IEEE Trans. Information Theory, 38(2), 644-664, March 1992

Depalle 90 — **Ph. Depalle and X. Rodet**, *Synthèse additive par FFT inverse*, Rapport Interne IRCAM, Paris, 1990

Depalle 93a — **Ph. Depalle, G. García, and X. Rodet**, *Analysis of Sound for Additive Synthesis: Tracking of Partials Using Hidden Markov Models*, Proc. ICMC, Tokyo, 94-97, 1993

Depalle 93b — **Ph. Depalle, G. García, and X. Rodet**, *Tracking of partials for additive sound synthesis using hidden Markov models*, Proc. ICASSP, Minneapolis, April 1993

De Poli 83 **G. De Poli**, *A Tutorial on Digital Sound Synthesis Techniques*, CMJ, 7(4), Winter 1983, also in The Music Machine – Selected Readings from Computer Music Journal, C. Roads, ed., Cambridge, Massachusetts: MIT Press, 1989

De Poli 93 **G. De Poli and P. Tonella**, *Self-organizing Neural Network and Grey's Timbre Space*, Proc. ICMC, Tokyo, 260-263, 1993

Desain 86 **P. Desain**, *Graphical Programming in Computer Music, a Proposal*, Proc. ICMC, The Hague, 161-166, 1986

Desain 93 **P. Desain and T. Brus**, *What Ever Happened To Our Beautiful Schematics*, Proc. ICMC, Tokyo, 366-368, 1993

Desain 94 **P. Desain and H. Honing**, *Foot-tapping: a brief introduction to beat induction*, Proc. ICMC, Aarhus, 78-79, 1994

de Tintis **R. de Tintis**, *Grains: a software for real-time gramular synthesis and sampling running on the IRIS-MARS workstation*, Proc. XI Colloquium on Musical Informatics, Bologna, 221-224, 1995

Deutsch **D. Deutsch**, *Grouping Mechanisms in Music*, in The Psychology of Music, D. Deutsch, ed., 99-130, New York: Academic Press, 1982

Di Giugno **G. Di Giugno**, *A 256 Digital Oscillator Bank*, Proc. ICMC, Massachusetts, 1976

Di Scipio **A. Di Scipio**, *Real-time Polyphonic Time-shifting of Sound with Interactive Systems*, Proc. XI Colloquium on Musical Informatics, Bologna, 19-22, 1995

Dixon **S. Dixon**, *A Dynamic Modelling Approach to Music Recognition*, Proc. ICMC, Hong Kong, 83-86, 1996

Dolson 86 **M. Dolson**, *The phase vocoder: a tutorial*, CMJ, 10(4), 14-27, Winter 1986

Dolson 91 **M. Dolson**, *Fourier-Transform-Based Timbral Manipulations*, in Current Directions in Computer Music Research, M.V. Mathews and J.R. Pierce, eds., 185-198, Cambridge, Massachusetts: MIT Press, 1991

Dorfman **L. Dorfman and D. Young**, *Atari ST: Introduction to MIDI Programming*, New York: Bantam

Doval **B. Doval and X.O. Rodet**, *Fundamental Frequency Estimation using a New Harmonic Matching Method*, Proc. ICMC, Montreal, 555-558, 1991

Dowling **W.J. Dowling**, *Melodic Information Processing and its Development*, in The Psychology of Music, D. Deutsch, ed., 413-429, New York: Academic Press, 1982

Dubnov **S. Dubnov, N. Tishby, and D. Cohen**, *Hearing Beyond the Spectrum*, JNMR, 24, 342-368, 1995

Duessenberry **J. Duessenberry**, *Understanding Amplitude Modulation*, Electronic Musician, November 1990

Eaglestone **B. Eaglestone and S. Oates**, *Analytical Tools for Group Additive Synthesis*, Proc. ICMC, Glasgow, 66-68, 1990

Edwards **T. Edwards**, *Discrete Wavelet Transforms: Theory and Implementation*, <ftp:// isl.stanford.edu/ pub/ godfrey/ reports/ wavelets/ tim_edwards/ wave_paper/ wave_paper.ps>, 9 July 1992

Elliott **D.F. Elliott and K.R. Rao**, *Fast transforms – algorithms, analyses, applications*, Academic Press, Orlando, Florida, 1982

Ellis 91 **D.P.W. Ellis and B.L. Vercoe**, *A wavelet-based sinusoid model of sound for auditory signal separation*, Proc. ICMC, Montreal, 86-89, 1991

Ellis 92a **D.P.W. Ellis and B.L. Vercoe**, *A Perceptual Representation of Sound for Auditory Signal Separation*, Proc. 123rd meeting of ASA, Salt Lake City, May 1992, and <ftp://sound.media.mit.edu/pub/Papers/dpwe-asa92slc.ps.gz>

Ellis 92b **D.P.W. Ellis**, *Timescale modifications and wavelet representations*, Proc. ICMC, San Jose, 6-9, 1992

Ellis 92c **D.P.W. Ellis**, *A Perceptual Representation of Audio*, MSc dissertation, Dept. of Elec. Engg. and Comp. Sci., MIT, 1992

Ellis 93 **D.P.W. Ellis**, *Hierarchical models of hearing for sound separation and reconstruction*, Proc. IEEE WASPAA, Mohonk, October 1993, and MIT Media Lab Perceptual Computing Group Tech. Report #219

Ellis 94 **D.P.W. Ellis**, *A Computer Implementation of Psychoacoustic Grouping Rules*, Proc. 12th Int. Conf. on Pattern Recognition, Jerusalem, October 1994, and MIT Media Lab Perceptual Computing Group Tech. Report #224 (rev. 2), and <ftp://sound.media.mit.edu/pub/Papers/dpwe-ICPR94.ps.gz>

Ellis 95a **D.P.W. Ellis and D. Rosenthal**, *Mid-level representations for Computational Auditory Scene Analysis*, Proc. Computational Auditory Scene Analysis Workshop, Int. Joint Conf. on Artificial Intelligence, Montreal, August 1995

Ellis 95b **D.P.W. Ellis**, *Underconstrained stochastic representations for top-down computational auditory scene analysis*, IEEE Workshop on Applications of Signal Processing to Audio and Acoustics, 1995

Ellis 96a **D.P.W. Ellis**, *Prediction-driven Computational Auditory Scene Analysis*, PhD thesis, Dept. of Elec. Eng. and Comp. Sci., MIT, 1996

Ellis 96b **D.P.W. Ellis**, *Prediction-driven Computational Auditory Scene Analysis for Dense Sound Mixtures*, to be presented at the ESCA workshop on the Auditory Basis of Speech Perception, Keele, UK, July 1996

Ellis 96c **D.P.W. Ellis**, *Hard Problems in Computational Auditory Scene Analysis for Dense Sound Mixtures*, <http://sound.media.mit.edu/~dpwe/writing/hard-probs.html>, 1996

Emagic **Emagic Inc.**, *Logic Audio (sales literature)*, <http://www.emagic.de/english/products/windows/LOGICAUDIO.html>, 1997

Erbe **T. Erbe**, *SoundHack User's Manual*, Mills College, Oakland, May 1992

Ethington **R. Ethington and B. Punch**, *SeaWave: A System for Musical Timbre Description*, CMJ, 18(1), 30-39, Spring 1994

Feiten 90 **B. Feiten and T. Ungvary**, *Sound data base using spectral analysis reduction and an additive synthesis model*, Proc. ICMC, Glasgow, 72-74, 1990

Feiten 91 **B. Feiten, R. Frank, and T. Ungvary**, *Organizations of sounds with neural nets*, Proc. ICMC, Montreal, 441-444, 1991

Feldman **M. Feldman** <myndale@cairo.anu.edu.au>, *PC Games Programmer's Encyclopaedia*, <http://www.qzx.com/pc-gpe/>, May 1994

Fernández-Cid **P. Fernández-Cid and F.J. Casajús-Quirós**, *DSP based reliable pitch-to-MIDI converter by harmonic matching*, Proc. ICMC, Aarhus, 307-310, 1994

Fitz — **K. Fitz, W. Walker, and L. Haken**, *Extending the McAulay-Quatieri Analysis for Synthesis with a Limited Number of Oscillators*, Proc. ICMC, Banff, 381-382, 1995

Fletcher 62 — **H. Fletcher, E.D. Blackham, and R. Stratton**, *Quality of piano tones*, JASA, 34, 749-761, 1962

Fletcher 63 — **H. Fletcher, E.D. Blackham, and D.A. Christensen**, *Quality of organ tones*, JASA, 35, 314-325, 1963

Fletcher 77a — **H. Fletcher**, *Normal Vibration Frequencies of a Stiff Piano String*, in Musical Acoustics – Piano and Wind Instruments, E.L. Kent, ed., Stroudsburg: Dowden, Hutchinson and Ross, 1977

Fletcher 77b — **H. Fletcher**, *Quality of Piano Tones*, in Musical Acoustics – Piano and Wind Instruments, E.L. Kent, ed., Stroudsburg: Dowden, Hutchinson and Ross, 1977

Fontana — **F. Fontana and D. Rochesso**, *A New Formulation of the 2D-Waveguide Mesh for Percussion Instruments*, Proc. XI Colloquium on Musical Informatics, Bologna, 27-30, 1995

Foster — **S. Foster, W.A. Schloss, and A.J. Rockmore**, *Toward an Intelligent Editor of Digital Audio: Signal Processing Methods*, CMJ, 6(1), 42-51, Spring 1982

Frandsen — **J. Frandsen** <jesperf@daimi.aau.dk>, *PCXDUMP*, <http://www.daimi.aau.dk/~jesperf/Pcxdump/pcxdump.html>, February 1994

Freed 93a — **A. Freed, X. Rodet, and Ph. Depalle**, *Synthesis and Control of Hundreds of Sinusoidal Partials on a Desktop Computer without Custom Hardware*, Proc. ICMC, Tokyo, 98-101, 1993

Freed 93b — **A. Freed, X. Rodet, and Ph. Depalle**, *Synthesis and Control of Hundreds of Sinusoidal Partials on a Desktop Computer without Custom Hardware*, Proc. ICSPAT, 1993

Gabor 46 — **D. Gabor,** *Theory of communication*, J. Inst. Elec. Engg., 93(III), 429-457, 1946

Gabor 47 — **D. Gabor,** *Acoustical quanta and the theory of hearing*, Nature, 159(4044), 1947

Goldstein 67 — **J.L. Goldstein**, *Auditory Nonlinearity*, JASA, 41, 676-689, 1967

Goldstein 73 — **J.L. Goldstein**, *An optimum processor for the central formation of pitch of complex tones*, JASA, 54(6), 1496-1516, 1973

Goodwin — **M. Goodwin and A. Kogon**, *Overlap-add synthesis of nonstationary simusoids*, Proc. ICMC, Banff, 355-356, 1995

Goto 94 — **M. Goto and Y. Muraoka**, *A Sound Source Separation System for Percussion Instruments*, Trans. Institute of Electronics, Information and Communication Engineers, J77-D-II(5), 901-911, May 94 [in Japanese]

Goto 96 — **M. Goto, I. Hidaka, H. Matsumoto, Y. Kuroda, and Y. Muraoka**, *A Jazz Session System for Interplay among All Players*, Proc. ICMC, Hong Kong, 346-349, 1996

Gozum — **L. Gozum and M. Gozum**, *VGACAP and VGAFIL (documentation)*, <ftp://sunsite.doc.ic.ac.uk/packages/simtelnet/msdos/vga/vgacap81.zip>, 1991

Graps — **A. Graps**, *An Introduction to Wavelets*, IEEE Computational Science and Engineering, 2(2), 50-61, Summer 1995

Gravis — **Advanced Gravis Computer Technology Ltd.**, *UltraSound User's Guide*, 1994

Gray — **J.M. Gray,** *Phonemic microtomy: The minimum duration of perceptible speech sounds*, Speech Monographs, 9, 75-90, 1942

GreenD — **D.M. Green**, *Temporal Auditory Acuity*, Psychological Review, 78(6), 540-551, 1971

GreenG — **G.D. Green,** <greeng@crl.com>, *"Contrabass saxes"*, private email, July 1996, and CONTRABASS-L mailing list, 1(45), <http://www.contrabass.com/pages/list.html>, 4 October 1996

Grey 75 — **J.M. Grey**, *An Exploration of Musical Timbre: using Computer-based Techniques for Analysis, Synthesis and Perceptual Scaling*, PhD thesis, Stanford Univ., and Stanford Music Dept. Technical Report STAN-M-2, 1975

Grey 77a — **J.M. Grey**, *Multidimensional Perceptual Scaling of Musical Timbres*, JASA, 61(5), 1270-1277, May 1977

Grey 77b — **J.M. Grey and J.A. Moorer**, *Perceptual Evaluation of Synthesized Musical Instrument Tones*, JASA, 62(2), 454-462, August 1977

Gribonval — **R. Gribonval, Ph. Depalle, X. Rodet, E. Bacry, and S. Mallat**, *Sound Signals Decomposition Using a High Resolution Matching Pursuit*, Proc. ICMC, Hong Kong, 293-296, 1996

Grieg 68 — **E. Grieg**, *Piano Concerto in A minor, Op. 16*, 1868 (audio: Compact Disc CLS4013, Düsseldorf: Mediaphon) (score: Leipzig: C.F. Peters)

Grieg 76 — **E. Grieg**, *Death of Aase, Peer Gynt Suite No. 1, Op. 46*, 1876 (audio: Compact Disc CLS4013, Düsseldorf: Mediaphon) (score: London: Eulenburg)

Grubb — **L. Grubb and R.B. Dannenberg**, *Automating Ensemble Performance*, Proc. ICMC, Aarhus, 63-69, 1994

Guillemain — **Ph. Guillemain and R. Kronland-Martinet**, *Additive resynthesis of sounds using continuous time-frequency representations*, Proc. ICMC, San Jose, 10-13, 1992

Haken 89 — **L. Haken**, *Real-Time Fourier Synthesis of Ensembles with Timbral Interpolation*, PhD thesis, Dept. of Electrical and Computer Engineering, Univ. of Illinois, 1989

Haken 92 — **L. Haken**, *Computational Methods for Real-Time Fourier Synthesis*, IEEE Trans. ASSP, 40(9), 2327-2329, 1992

Hall — **G. Hall**, *The Dimensions of Delay*, Electronic Musician, September 1990

Handel — **S. Handel**, *Listening*, Cambridge, Massachusetts: MIT Press, 1989

Hanson — **B.A. Hanson and D.Y. Wong**, *The Harmonic Magnitude Suppression Technique for Intelligibility Enhancement in the Presence of Interfering Speech*, Proc. ICASSP, 18A.5.1-18A.5.4, 1984

Harris — **F. J. Harris**, *On the use of windows for harmonic analysis with the DFT*, Proc. IEEE 66, 51-83, 1978

Hawley — **M. Hawley**, *Structure out of Sound*, PhD thesis, MIT Media Lab., 1993

Heinbach 87 — **W. Heinbach**, *Gehörgerecht Repräsentation von Audiosignalen durch das Teiltonzeitmuster*, PhD thesis, Technical University of Munich, 1987

Heinbach 88 — **W. Heinbach**, *Aurally adequate signal representation: The part-time-tone-pattern*, Acustica, 67, 113-121, 1988

Helmholtz — **H.L.F. von Helmholtz**, *On the Sensations of Tome as a Physiological Basis for the Theory of Music*, 1863, 4th ed., trans. A.J. Ellis, New York: Dover, 1954

Helmuth 93 — **M. Helmuth**, *Granular Synthesis with Cmix and MAX*, Proc. ICMC, Tokyo, 449-452, 1993

Helmuth 96 — **M. Helmuth**, *Multidimensional Representation of Electroacoustic Music*, JNMR, 25, 77-103, 1996

Hermes — **D.J. Hermes**, *Measurement of pitch by harmonic summation*, JASA, 83(1), 257-264, 1988

Hesseler — **W. Hesseler** <hesseler@athene.informatik.uni-bonn.de>, *QuickView*, <ftp://ftp.rhrz.uni-bonn.de/pub/institute/hesseler/qv103a.zip>, 1995

Hidaka — **I. Hidaka, M. Goto, and Y. Muraoka**, *An Automatic Jazz Accompaniment System Reacting to Solo*, Proc. ICMC, Banff, 167-170, 1995

Hirsh — **I.J. Hirsh**, *Auditory Perception of Order*, JASA, 39, 759, 1959

Hohner — **Hohner Midia,** *Soundscore (program)*, <http://www.hohnermidia.com/basic.html>, September 1996

Höldrich 94a — **R.R. Höldrich**, *Zur Analyse und Resynthese von Klangsignalen unter Verwendung von Zeit-Frequenze-Repräsentationen mit Verbesserter Lokalisation der Signalenergie*, PhD thesis, Technical Univ. Graz, Austria, 1994 [in German]

Höldrich 94b — **R.R. Höldrich**, *The Improved Spectrogram – How to Obtain an Accurate Signal Representation for Sound Resynthesis?*, Proc. Int. Simposio Brasiliera de Computacao e Musica, 1994

Höldrich 94c — **R.R. Höldrich**, *Frequency Analysis of Non-Stationary Signals Using a Time-Frequency Mapping of the DFT Magnitude*, Proc. ICSPAT, Dallas, 1994

Höldrich 95 — **R.R. Höldrich**, *An Accurate Signal Representation for Sound Resynthesis Utilizing a Time-Frequency Mapping of the DFT-Magnitude*, Proc. ICMC, Banff, 592-594, 1995

Horiuchi 92 — **Y. Horiuchi, A. Fujii, and H. Tanaka**, *A Computer Accompaniment System Considering Independence of Accompanist*, Japan Music and Computer Science Society, Proc. of Summer Symposium, 1992 [in Japanese]

Horiuchi 93 — **Y. Horiuchi and H. Tanaka**, *A Computer Accompaniment System With Independence*, Proc. ICMC, Tokyo, 418-420, 1993

Horner 93 — **A. Horner, J. Beauchamp, and N. Packard**, *Timbre Breeding*, Proc. ICMC, Tokyo, 396-398, 1993

Horner 95a — **A. Horner**, *Envelope Matching with Genetic Algorithms*, JNMR, 24, 318-341, 1995

Horner 95b — **A. Horner**, *Wavetable Matching Synthesis of Dynamic Instruments with Genetic Algorithms*, J. AES, 43(11), 916-931, 1995

Horner 96 — **A. Horner and J. Beauchamp**, *Piecewise-Linear Approximation of Additive Synthesis Envelopes: A Comparison of Various Methods*, CMJ, 20(2), 72-95, Summer 1996

Horry — **Y. Horry**, *A Graphical User Interface for MIDI Signal Generation and Sound Synthesis,* Proc. ICMC, Aarhus, 276-279, 1994

Houghton — **A.D. Houghton, A.J. Fisher, and T.F. Malet**, *An ASIC for Digital Additive Sine-wave Synthesis,* CMJ, 19(3), 26-31, Fall 1995

Hu — **Y.H. Hu**, *CORDIC-Based VLSI Architectures for Digital Sound Processing,* IEEE Signal Processing Magazine, July 1992

Hughes — **R.D. Hughes and M.L. Heron**, *Approximate Fourier transform using square waves,* Proc. IEE, 136(A4), 223-227, July 1989

Hung — **R. Hung, N.H.C. Yung, and P.Y.S. Cheung**, *The Analysis and Resynthesis of Sustained Musical Signals in the Time Domain,* Proc. ICMC, Hong Kong, 206-209, 1996

Hyun — **K.R. Hyun, R. Banerjea, M. Kim, H. Latchman, and S.I. Sudharsanan,** *A Real-Time Implementation of MPEG Audio Layer I Decoding on a Fixed-Point DSP Platform,* Proc. ICMC, Tokyo, 412-414, 1993

Inoue 93 — **W. Inoue, S. Hashimoto, and S. Ohteru,** *A Computer Music System for Human Singing,* Proc. ICMC, Tokyo, 150-153, 1993

Inoue 94 — **W. Inoue, S. Hashimoto, and S. Ohteru,** *Adaptive Karaoke System – Human Singing Accompaniment Based on Speech Recognition,* Proc. ICMC, Aarhus, 70-77, 1994

ISO — **ISO/IECJTC1/SC2/WG11 MPEG 91 Committee Draft**, *Coding of moving pictures and Associated audio for digital storage media at up to 1.5 Mbit/s,* November 1991

Itagaki 94 — **T. Itagaki, A. Purvis, and P.D. Manning**, *Real-time Synthesis on a Multi-processor network,* Proc. ICMC, Aarhus, 382-385, 1994

Itagaki 95a — **T. Itagaki, D.K. Phillips, P.D. Manning, and A. Purvis**, *An Implementation of Optimised Methods for Real-time Sound Synthesis on a Multi-processor Network,* Book of Abstracts on Parallel Computing, Gent, 1995

Itagaki 95b — **T. Itagaki, D.J.E. Nunn, D.K. Phillips, D. Batjakis, A. Purvis, and P.D. Manning**, *Activity Report,* XI Colloquium on Musical Informatics, Bologna, 51-54, 1995

Itagaki 96a — **T. Itagaki, P.D. Manning, and A. Purvis,** *Real-time Gramular Synthesis on a Distributed Multi-processor Platform,* Proc. ICMC, Hong Kong, 287-288, 1996

Itagaki 96b — **T. Itagaki, S. Johnson, P.D. Manning, D.J.E. Nunn, D.K. Phillips, A. Purvis, and J. Spanier**, *Durham Music Technology: Activity Report,* Proc. ICMC, Hong Kong, 126-128, 1996

Jaffe — **D.A. Jaffe**, *Ten Criteria for Evaluating Synthesis Techniques*, CMJ, 19(1), 76-87, Spring 1995

Jain — **A.K. Jain**, *A fast Karhunen-Loêve transform for a class of random processes*, IEEE Trans. Commun., COM-24, 1023-1029, 1976

Jánosy — **Z. Jánosy, M. Karjalainen, and V. Välimäki**, *Intelligent Synthesis Control with Applications to a Physical Model of the Acoustic Guitar*, Proc. ICMC, Aarhus, 402-406, 1994

Jansen 91 **C. Jansen**, *Sine Circuitu – 10,000 high quality sine waves without detours*, Proc. ICMC, Montreal, 222-225, 1991

Jansen 92 **C. Jansen**, *Sine Circuitu – Real-time analysis, manipulation and (re)synthesis*, Proc. ICMC, San Jose, 451-452, 1992

Jaw **S.-B. Jaw and S.-C. Pei**, *Two-band IIR quadrature mirror filter design*, Electronics Letters, 26(20), 1687-1689, September 1990

Jawerth **B. Jawerth and W. Sweldens**, *An overview of wavelet based multiresolution analysis*, SIAM Rev., 36(3), 377-412, 1994, and <http://cm.bell-labs.com/who/wim/papers/overview.ps.gz>

Jones 87 **D.L. Jones and T.W. Parks**, *On computing equally spaced samples of a complex Gaussian function*, IEEE Trans. ASSP, 35(10), 1987

Jones 88 **D.L. Jones and T.W. Parks**, *Generation and combination of grains for music synthesis*, CMJ, 12(2), 27-34, Summer 1988

Jordan **Jordan Hargraphics Software Inc.**, *SVGABGI*, <gopher://micros.hensa.ac.uk:70/11/micros/ibmpc/dos/g/g770/>, January 1995

Justice **J.H. Justice**, *Analytic Signal Processing in Music Computation*, IEEE Trans. ASSP, 27(6), 670-684, 1979

Kadambe **S. Kadambe and G.F. Boudreaux-Bartels**, *Application of the Wavelet Transform for Pitch Detection of Speech Signals*, IEEE Trans. Information Theory, 38(2), 917-924, 1992

Kaiser 74 **J.F. Kaiser**, *Nonrecursive digital filter design using the I0-sinh window function*, Proc. IEEE Int. Symposium on Circuits and Systems, 20-23, San Francisco, 1974

Kaiser 87 **J.F. Kaiser**, *On the fast generation of equally spaced values of the Gaussian function A exp(-at*t)*, IEEE Trans. ASSP, 35(10), 1987

Karjalainen 93 **M. Karjalainen, V. Välimäki, and Z. Jánosy**, *Towards High-Quality Sound Synthesis of the Guitar and String Instruments*, Proc. ICMC, Tokyo, 56-63, 1993

Karjalainen 96 **M. Karjalainen and J. Smith,** *Body Modeling Techniques for String Instrument Synthesis,* Proc. ICMC, Hong Kong, 232-239, 1996

Kashino 92 **K. Kashino and H. Tanaka**, *A sound source separation system using spectral features integrated by [the] Dempster's law of combination*, In Annual Report of the Engineering Research Institute, Faculty of Engineering, University of Tokyo, 67-72, 1992

Kashino 93 **K. Kashino and H. Tanaka**, *A Sound Source Separation System with the Ability of Automatic Tone Modelling*, Proc. ICMC, Tokyo, 248-255, 1993

Kashino 95a **K. Kashino, K. Nakadai, T. Kinoshita, and H. Tanaka**, *Organization of Hierarchical Perceptual Sounds: Music Scene Analysis with Autonomous Processing Modules and a Quantitative Information Integration Mechanism*, Proc. Int. Joint Conf. on Artificial Intelligence, Workshop on Computational Auditory Scene Analysis, Montreal, August 1995, and <http://www.mtl.t.u-tokyo.ac.jp/Research/paper/1995/E95-conference-kashino-1.ps.gz>

Kashino 95b **K. Kashino, K. Nakadai, T. Kinoshita, and H. Tanaka**, *Application of Bayesian Probability Network to Music Scene Analysis*, Proc. Int. Joint Conf. on Artificial Intelligence, Workshop on Computational Auditory Scene

Analysis, Montreal, August 1995, and <http://www.mtl.t.u-tokyo.ac.jp/Research/paper/1995/E95-conference-kashino-2.ps.gz>

Katayose 88 — **H. Katayose, M. Imai, and S. Inokuchi**, *Sentiment Extraction in Music*, Proc. 9th Int. Conf. on Pattern Recognition, 1083-1087, Washington, 1988

Katayose 89 — **H. Katayose and S. Inokuchi**, *The Kansei Music System*, CMJ, 13(4), 72-77, Winter 1989

Katayose 90a — **H. Katayose and S. Inokuchi**, *The Kansei Music System '90*, Proc. ICMC, Glasgow, 308-310, 1990

Katayose 90b — **H. Katayose, T. Fukuoka, K. Takami, and S. Inokuchi**, *Expression extraction in virtuoso music performances*, Proc. 10th Int. Conf. on Pattern Recognition, New Jersey, 780-784, June 1990

Katayose 93 — **H. Katayose, T. Kanamori, K. Kamei, Y. Nagashima, K. Sato, S. Inokuchi, and S. Simura**, *Virtual Performer*, Proc. ICMC, Tokyo, 138-145, 1993

Kendall 91 — **G.S Kendall, W.L. Martens, and S.L. Decker**, *Spatial Reverberation: Discussion and Demonstration*, in Current Directions in Computer Music Research, M.V. Mathews and J.R. Pierce, eds., 65-87, Cambridge, Massachusetts: MIT Press, 1991

Kendall 95 — **G.S. Kendall**, *A 3-D Sound Primer: Directional Hearing and Stereo Reproduction*, CMJ, 19(4), 23-46, Winter 1995

Kernighan — **B.W. Kernighan and D.M. Ritchie**, *The C Programming Language*, London: Prentice-Hall, 1988

Kiang — **N.Y.S. Kiang and E.C. Moxon**, *Tails of tuning curves of auditory-nerve fibers*, JASA, 55(3), 620-630, 1974

Kleczkowski — **P. Kleczkowski**, *Group Additive Synthesis*, CMJ, 13(1), 12-20, Spring 1989

Kodera 76 — **K. Kodera**, *Analyse numérique de signeaux géophysiques nonstationaires*, PhD thesis, Univ. of Paris, 1976 [in French]

Kodera 78 — **K. Kodera, R. Gendrin, and C. Villedary**, *Analysis of Time-Varying Signals with Small BT Values*, IEEE Trans. ASSP, 26(1), 64-76, 1978

Kohonen — **T. Kohonen**, *The Self-Organizing Map*, Proc. IEEE, 78(9), 1464-1480, 1990

Kottick — **E.L. Kottick, K.D. Marshall, and T.J. Hendrickson**, *The Acoustics of the Harpsichord*, Scientific American, 94-99, February 1991

Kriese — **C. Kriese and S. Tipei**, *A compositional approach to additive synthesis on supercomputers*, Proc. ICMC, San Jose, 394-395, 1992

Kronland-Martinet 87 — **R. Kronland-Martinet, J. Morlet, and A. Grossmann**, *Analysis of sound patterns through wavelet transforms*, Int. J. of Pattern Recognition and Artificial Intelligence, 2, 97-126, 1987

Kronland-Martinet 88 — **R. Kronland-Martinet,** *The Wavelet Transform for Analysis, Synthesis, and Processing of Speech and Music Sounds*, CMJ, 12(4), 11-20, Winter 1988

Kronland-Martinet 93 — **R. Kronland-Martinet and Ph. Guillemain**, *Towards non-linear resynthesis of instrumental sounds*, Proc. ICMC, Tokyo, 86-93, 1993

Kuhn — **W.B. Kuhn,** *A Real-Time Pitch Recognition Algorithm for Music Applications,* CMJ, 14(3), 60-71, Fall 1990

Kurz — **M. Kurz and B. Feiten**, *Physical modelling of a stiff string by numeric integration*, Proc. ICMC, Hong Kong, 361-364, 1996

Kussmaul — **C. Kussmaul**, *Applications of the Wavelet Transform at the Level of Pitch Contour*, Proc. ICMC, Glasgow, 483-486, 1990

Laden 91 — **B. Laden and D.H. Keefe**, *The Representation of Pitch in a Neural Net Model of Chord Classification*, in Music and Connectionism, P.M. Todd and D.G. Loy, eds., Cambridge, Massachusetts: MIT Press, 1991

Laden 94 — **B. Laden**, *A Parallel Learning Model of Musical Pitch Perception*, JNMR, 23, 133-144, June 1994

Lane — **J.E. Lane**, *Pitch Detection Using a Tunable IIR Filter*, CMJ, 14(3), 46-59, Fall 1990

Leman — **M. Leman**, *Schema-Based Tone Center Recognition of Musical Signals*, JNMR, 23(2), 169-204, 1994

Licklider — **J.C.R. Licklider**, *Three Auditory Theories*, in Psychology: A Study in Science, Vol. 1, S. Koch, ed., New York: McGraw-Hill, 1959

Lindemann 90 — **E. Lindemann, M. Starkier, and F. Dechelle,** *The IRCAM Musical Workstation: Hardware Overview and Signal Processing Features*, Proc. ICMC, Glasgow, 132-135, 1990

Lindemann 91 — **E. Lindemann, F. Dechelle, B. Smith, and M. Starkier,** *The Architecture of The IRCAM Musical Workstation*, CMJ, 15(3), 41-49, 1991

Lippe 91 — **C. Lippe and M. Puckette,** *Musical Performance using the IRCAM Workstation*, Proc. ICMC, Montreal, 533-536, 1991

Lippe 93a — **C. Lippe,** *A Musical Application of Real-time Granular Sampling Using the IRCAM Signal Processing Workstation*, Proc. ICMC, Tokyo, 190-193, 1993

Lippe 93b — **C. Lippe, M. Puckette, Z. Settel, V. Puig, and J-P. Jullien,** *The IRCAM Signal Processing Workstation and IRCAM Max User Groups: Future Developments and Platforms*, Proc. ICMC, Tokyo, 446-448, 1993

Lo — **D. Y-O. Lo,** *Techniques for Timbral Interpolation*, Proc. ICMC, The Hague, 1986

Lopez-Lezcano — **F. Lopez-Lezcano,** *PadMaster: banging on algorithms with alternative controllers*, Proc. ICMC, Hong Kong, 425-427, 1996

Loy — **G. Loy,** *Composing with Computers – a Survey of Some Compositional Formalisms and Music Programming Languages*, in Current Directions in Computer Music Research, M.V. Mathews and J.R. Pierce, eds., 291-396, Cambridge, Massachusetts: MIT Press, 1991

LTSound — **LT Sound, Inc.,** *The Thompson Vocal Eliminator*, sales literature, 1996

Maggi — **E. Maggi and F. Dechelle**, *The evolution of the graphic editing environment for the IRCAM musical workstation*, Proc. ICMC, Hong Kong, 185-187, 1996

Maher 89 — **R.C. Maher**, *An Approach for the Separation of Voices in Composite Musical Signals*, PhD thesis, University of Illinois, Urbana, 1989

Maher 90 — **R.C. Maher**, *Evaluation of a Method for Separating Digitized Duet Signals*, J. AES, 38(12), 956-979, 1990

Majerník — **V. Majerník and J. Kaluzný**, *On the Auditory Uncertainty Relations*, Acustica, 43, 132, 1979

Maksym — **J.N. Maksym**, *Real-time pitch extraction by adaptive prediction of the speech waveform*, IEEE Trans. on Audio and Electroacoustics, 21, 149-154, 1973

Mallat 89 — **S.G. Mallat**, *A Theory for Multiresolution Signal Decomposition: The Wavelet Representation*, IEEE Trans. on Pattern Analysis and Machine Intelligence, 11(7), 674-693, July 1989

Mallat 93 — **S.G. Mallat and Z. Zhang**, *Matching Pursuit with Time-Frequency Dictionaries*, IEEE Trans. on Signal Processing, 41, 3397-3415, 1993

Manning — **P.D. Manning**, *Electronic and Computer Music*, Oxford: Oxford University Press, 1985

Mansour — **A. Mansour and C. Jutten**, *A Simple Cost Function For Instantaneous and Convolutive Sources Separation*, Actes du XVème Colloque GRETSI, 301-304, September 1995

Markel — **J.D. Markel and A.H. Gray**, *Linear Prediction of Speech*, Berlin: Springer-Verlag, 1976

Marple — **S.L. Marple, Jr.**, *Digital Spectral Analysis with Applications*, Englewood Cliffs: Prentice-Hall, 1987

MartinD — **D. Martin and D. Ward**, *Subjective evaluation of musical scale temperament in pianos*, JASA, 33, 582-585, 1961

MartinK 95 — **K.D. Martin**, *Estimating azimuth and elevation from interaural differences*, IEEE WASPAA, Mohonk 1995, and <ftp://sound.media.mit.edu/pub/Papers/kdm-mohonk95.ps.Z>

MartinK 96 — **K.D. Martin**, *Blackboard-Based Transcription Project*, <http://sound.media.mit.edu/~kdm/research/bboard/>, April 1996

Mason — **D.K. Mason** <76546.1321@compuserve.com>, *'Dave's Targa Animator' and 'Dave's Flic Viewer' programs*, <http://www.povray.org/povcd/programs/povutil/dfv_dta/dta21pb.zip>, 1995

Mathews 69 — **M.V. Mathews**, *The Technology of Computer Music*, Cambridge, Massachusetts: MIT Press, 1969

Mathews 91 — **M.V. Mathews and J.R. Pierce**, *The Bohlen-Pierce Scale*, in Current Directions in Computer Music Research, M.V. Mathews and J.R. Pierce, eds., 165-173, Cambridge, Massachusetts: MIT Press, 1991

MathSoft — **MathSoft Inc.**, *MathCad*, <http://www.mathsoft.com/all60.htm>, 1994

Matthews — **P. Matthews and N. McWhirter**, *The Guinness Book of Records*, London: Guinness Publishing, 1995

McAdams — **S.J. McAdams**, *Spectral Fusion, Spectral Parsing and the Formation of Auditory Images*, PhD thesis, Stanford Univ., 1984

McAulay — **R.J. McAulay and T.E. Quatieri**, *Speech analysis/synthesis based on a sinusoidal representation*, Trans. ASSP, ASSP-34, 744-754, 1986

Mellinger 91a — **D.K. Mellinger**, *Event formation and separation in musical sound*, PhD thesis, Stanford Univ., 1991

Mellinger 91b — **D.K. Mellinger and B.M. Mont-Reynaud**, *SoundExplorer: A Workbench for Investigating Source Separation*, Proc. ICMC, Montreal, 90-93, 1991

Mendelssohn — **F. Mendelssohn**, *Sonata III, from Sechs Sonaten für die Orgel, Op. 65*, 1844 (score: Leipzig: Breitkopf & Härtel, republished (1967) Farnborough, UK: Gregg Press, 1967)

Menninga — **S. Menninga**, *"WAV -> MIDI? ... about Physics and feasibility"*, private email, 7 November 1996

Meyer — **Y. Meyer**, *Wavelets: Examples and Applications*, SIAM, Philadelphia, 1993

Microsoft — **Microsoft Inc.**, *QBasic (part of MS-DOS)*, 1993

Milicevic — **M. Milicevic**, *The Impact of Fractals, Chaos, and Complexity Theory on Computer Music Composition*, Proc. ICMC, Hong Kong, 473-476, 1996

Miller — **J.R. Miller and E.C. Carterette**, *Perceptual space for musical structures*, JASA, 58(3), 711-720, 1975

Mintzer — **F. Mintzer**, *Filters for Distortion-Free Two-Band Multirate Filter Banks*, IEEE Trans. ASSP, 33(3), 1985

Miranda — **E. Miranda**, *Cellular Automata Synthesis of Acoustic Particles*, <http:// www.epcc.ed.ac.uk/tracs/eduardo.html>, 11 May 1995

Mitchell — **O.M.M. Mitchell, C.A. Ross, and G.H. Yates**, *Signal processing for a cocktail party effect*, JASA, 50(2), 656,660, 1971

Mont-Reynaud 85 — **B. Mont-Reynaud**, *Problem-solving Strategies in a Music Transcription System*, Proc. Int. Joint Conf. on Artificial Intelligence, 916-918, 1985

Mont-Reynaud 88 — **B. Mont-Reynaud**, *On Hearing Music Visually*, Proc. AAAI, Special Session on Artificial Intelligence and Music, St. Paul, 1988

Mont-Reynaud 89 — **B. Mont-Reynaud and D. Mellinger**, *Source Separation by Frequency Co-Modulation*, Proc. of the First Int. Conf. on Music Perception and Cognition, Kyoto, 99-102, 1989

Mont-Reynaud 90 — **B. Mont-Reynaud and E. Gresset**, *PRISM: Pattern Recognition In Sound and Music*, Proc. ICMC, Glasgow, 153-155, 1990

Mont-Reynaud 93 — **B. Mont-Reynaud**, *SeeMusic: A Tool for Music Visualization*, Proc. ICMC, Tokyo, 457-460, 1993

MooreB 85 — **B.C.J. Moore, B. Glasberg, and R.W. Peters**, *Relative dominance of individual partials in determining the pitch of complex tones*, JASA, 77, 1853-1860, 1985

MooreB 95 — **B.C.J. Moore**, *Hearing*, San Diego: Academic Press, 1995

MooreD — **D.R. Moore**, *Physiology of higher auditory system*, British Journal of Audiology, 24, 131-137, 1987

MooreF 77 — **F.R. Moore**, *Table Lookup Noise for Sinusoidal Digital Oscillators*, CMJ, 1(2), 26-29, Summer 1977

MooreF 88 — **F.R. Moore**, *The Dysfunctions of MIDI*, CMJ, 12(1), 19-28, Spring 1988

MooreF 90 — **F.R. Moore**, *Elements of Computer Music*, Englewood Cliffs, Prentice-Hall, 1990

Moorer 75 — **J.A. Moorer**, *On the Segmentation and Analysis of Continuous Musical Sound by Digital Computer*, PhD thesis, Dept. of Computer Science, Stanford Univ., and Stanford Music Dept. Technical Report STAN-M-3, 1975

Moorer 76 — **J.A. Moorer**, *The Synthesis of Complex Audio Spectra by Means of Discrete Summation Formulas*, J. AES, 24(9), 717-727, November 1976

Moorer 77a — **J.A. Moorer**, *Signal Processing aspects of Computer Music: A Survey*, Proc. IEEE, 65, 1108-1137, 1977

Moorer 77b — **J.A. Moorer**, *On the Transcription of Musical Sound by Computer,* CMJ, 1(4), 32-38, Winter 1977

Moorer 78 — **J.A. Moorer**, *The use of the phase vocoder in computer music applications,* J. AES, 24(9), 1978

Moreno — **E.I. Moreno,** *The Existence of Unexplored Dimensions of Pitch: Expanded Chromas,* Proc. ICMC, San Jose, 402-403, 1992

Motorola — **Motorola Inc.,** *DSP96002 IEEE Floating-Point Dual-Port Processor User's Manual,* 1989

Nakatani — **T. Nakatani, H.G. Okuno, and T. Kawabata**, *Auditory stream segregation in auditory scene analysis with a multi-agent system,* AAAI Conference Proceedings, 1994

Naoi — **K. Naoi, S. Ohteru, and S. Hashimoto**, *Automatic Accompaniment Using Real Time Assigning Note Value,* Convention Record of the Acoustical Society of Japan, Spring 1989 [in Japanese]

Nawab — **S.H. Nawab and E. Dorken**, *Efficient STFT approximation using a quantization and differencing method,* Proc. ICASSP, Minneapolis, 587-590, April 1993

Naylor — **J.A. Naylor and S.F. Boll**, *Techniques for suppression of an Interfering Talker in Co-channel Speech,* Proc. ICASSP, 6.12.1-6.12.4, 1987

Neely — **S. Neely**, <neely@boystown.org>, *Cochlear Mechanics Tutorial,* Boys Town National Research Hospital, <http://www.boystown.org/cel/cochmech.html>, March 1995

Newland 93 — **D. Newland**, *Harmonic wavelet analysis,* Proc. Róyal Soc. London Series A – Mathematical and Physical Sciences, 443, 203-225, 1993

Newland 94 — **D. Newland**, *Harmonic and Musical Wavelets,* Proc. Royal Soc. London Series A – Mathematical and Physical Sciences, 444, 605-620, 1994

Nieberle — **R.C Nieberle**, *A Fast Communication Interface to the CAMP-DSP-Subsystem for general purpose Sound-Synthesis, Analysis and Processing,* Proc. ICMC, Montreal, 529-532, 1991

Niihara — **T. Niihara, H. Katayose, and S. Inokuchi**, *Transcription of Sung Song,* Proc. ICASSP, IEEE, New York, 1986

Noll — **A.M. Noll**, *Cepstrum Pitch Determination,* JASA, 41, 293-309, 1966

Nunn 84 — **D.J.E. Nunn,** *The Acoustics of Brass Instruments,* Sixth Year Studies Physics Project, George Watsons College, Edinburgh, 1984

Nunn 94 — **D.J.E. Nunn, A. Purvis, and P.D. Manning**, *Source Separation and Transcription of Polyphonic Music,* Proc. International Colloquium on New Music Research, Gent, Belgium, 1994

Nunn 95 — **D.J.E. Nunn, A. Purvis, and P.D. Manning**, *Graphical display of musical information,* Proc. XI Colloquium on Musical Informatics, Bologna, 235-236, 1995

Nunn 96 — **D.J.E. Nunn, A. Purvis, and P.D. Manning**, *Acoustic Quanta,* Proc. ICMC, Hong Kong, 52-54, 1996

Nuttall 81 — **A.H. Nuttall**, *Some windows with very good sidelobe behavior*, IEEE Trans. ASSP, 29, 84-87, 1981

Nuttall 87 — **A.H. Nuttall**, *Efficient Evaluation of Polynomials and Exponentials of Polynomials for Equispaced Arguments*, IEEE Trans. ASSP, 35(10), October 1987

Okamoto — **H. Okamoto,** *Mouchu (installation)*, ICMC Concert Program, Hong Kong, 85, 1996

Olson — **H. Olson**, *Music, Physics, and Engineering*, New York: Dover, 1967

Oohashi 89 — **T. Oohashi et al.**, *High frequency sound on "Trance Induction Music"*, Tech. Report on Musical Acoustics, Acoustical Society of Japan, ES88-77, 11-15, 1989

Oohashi 91 — **T. Oohashi et al.**, *High-frequency components above the audible range affects brain electric activity and sound perception*, AES 91st convention preprint 3207, 1991

Oohashi 93 — **T. Oohashi, E. Nishina, Y. Fuwamoto, and N. Kawai**, *On the Mechanism of "Hypersonic Effect"*, Proc. ICMC, Tokyo, 432-434, 1993

Opcode — **Opcode Systems**, *Studio Vision Pro (sales literature)*, <http:// www.opcode.com/products/svpro/>, 1996

Pabon 94a — **P. Pabon**, *A real-time singing voice analysis/synthesis system*, Proc. ICMC, Aarhus, 356, 1994

Pabon 94b — **P. Pabon**, *Real-time spectrum/cepstrum games*, Proc. ICMC, Aarhus, 361, 1994

Pacheco — **P.S. Pacheco**, *A User's Guide to MPI*, <ftp://math.usfca.edu/pub/MPI/ mpi.guide.ps.Z>, March 1995

Palmer — **C. Palmer**, *Timing in Skilled Musical Performance*, PhD thesis, Cornell Univ., 1989

Papoulis — **A. Papoulis**, *Signal Analysis*, New York: McGraw-Hill, 1987

Parash — **A. Parash and U. Shimony**, *An expandable real-time transputer sound generator*, Proc. ICMC, Montreal, 226-228, 1991

Parsons — **W. Parsons**, *Separation of speech from interfering speech by means of harmonic selection*, JASA, 60 (4), 911-918, 1976

PattersonB — **B. Patterson**, *Musical Dynamics*, Scientific American, 231(5): 78, 1974

PattersonR — **R.D. Patterson, J. Holdsworth, I. Nimmo-Smith, and P. Rice**, *SVOS final report: The gammatone filter bank*, Cambridge Applied Psychology Unit Report 2341, 1988

Pearson 90 — **E.R.S. Pearson and R.G. Wilson**, *Multiple Event Detection from Audio Signals within a Multiresolution Framework*, Proc. ICMC, Glasgow, 156-158, 1990

Pearson 91 — **E.R.S. Pearson**, *The multiresolution Fourier transform and its application to the analysis of polyphonic music*, PhD thesis, Warwick Univ., September 1991

Pennycook 86 — **B. Pennycook,** *Language and Resources: A New Paradox*, in The Language of Electroacoustic Music, S. Emmerson, ed., London: MacMillan Press, 1986

Pennycook 93 — **B. Pennycook, D.R. Stammen, and D. Reynolds**, *Toward a Computer Model of a Jazz Improviser*, Proc. ICMC, Tokyo, 228-231, 1993

Perez — **I. Perez**, *MacMUSIC, the MUSIC N environment for Macintosh, algorithmic synthesis and composition made easy*, Proc. ICMC, Aarhus, 239-240, 1994

Phillips 94 — **D.K. Phillips, A. Purvis, and S. Johnson**, *A Multirate Optimisation for Real-Time Additive Synthesis*, Proc. ICMC, Aarhus, 364-367, 1994

Phillips 96 — **D.K. Phillips, A. Purvis, and S. Johnson**, *Multirate Additive Synthesis*, Proc. ICMC, Hong Kong, 496-499, 1996

Pierce — **J.R. Pierce**, *Residues and Summation Tones – What Do We Hear?*, in Current Directions in Computer Music Research, M.V. Mathews and J.R. Pierce, eds., 175-184, Cambridge, Massachusetts: MIT Press, 1991

Piszczalski 77 — **M. Piszczalski and B.F. Galler**, *Automatic Music Transcription*, CMJ, 1(4), 24-31, Winter 1977

Piszczalski 79 — **M. Piszczalski and B.F. Galler**, *Predicting Musical Pitch from Component Frequency Ratios*, JASA, 66, 710-720, 1979

Piszczalski 81 — **M. Piszczalski, B.F. Galler, R. Bossmeyer, M. Hatamian, and F. Looft**, *Performed Music: Analysis, Synthesis, and Display by Computer*, J. AES, 29(1/2), 38-46, 1981

Pkware — **Pkware Inc.**, *PKZIP- documentation*, <http://www.pkware.com/pkzip.html>, 1996

Plomp 70 — **R. Plomp**, *Timbre as a Multidimensional Attribute of Complex Tones*, in Frequency Analysis and Periodicity Detection in Hearing, R. Plomp and G. Smoorenburg, eds., Leiden: Sijthoff, 1970

Plomp 76 — **R. Plomp**, *Aspects of tone sensation*, London: Academic Press, 1976

Pope — **S.T. Pope**, *Computer Music Workstations I Have Known and Loved*, Proc. ICMC, Banff, 127-133, 1995

Popovic 95a — **I. Popovic, R.R. Coifman, and J. Berger**, *Aspects of Pitch-Tracking and Timbre Separation: Feature Detection in Digital Audio Using Adapted Local Trigonometric Bases and Wavelet Packets*, Proc. ICMC, Banff, 280-283, 1995, and <http://www.music.yale.edu/research/pc/pitchtrack.html>

Popovic 95b — **I. Popovic, R.R. Coifman, and J. Berger**, *Toward a Unified Representation of Sound and Analytical Structure in Music*, Proc. XI Colloquium on Musical Informatics, Bologna, 55-58, 1995

Portnoff — **M.R. Portnoff**, *Time-scale modification of speech based on short-time Fourier analysis*, IEEE Trans. ASSP, 29(3), June 1981

Poulenc — **F. Poulenc**, *Sonata for Horn, Trumpet, and Trombone*, 1922. (audio: Record ZRG 731, London: Argo) (score: London: J.&W. Chester)

Press — **J. Press**, *Numerical Recipes: The Art of Scientific Computing*, Cambridge: Cambridge University Press, 1986, and <http://cfatab.harvard.edu/nr/nronline.html>, 1996

Pressing 93a — **J. Pressing and P. Lawrence**, *Transcribe: A Comprehensive Autotranscription Program*, Proc. ICMC, Tokyo, 343-345, 1993

Pressing 93b — **J. Pressing and P. Lawrence**, *Visualization and Predictive Modelling of Musical Signals using Embedding Techniques*, Proc. ICMC, Tokyo, 110-113, 1993

Pringle — **R. Pringle and B.J. Ross**, *A Symbiosis of Animation and Music*, Proc. ICMC, Hong Kong, 316-319, 1996

Prosoniq — **Prosoniq Inc.**, *The Timescale Modification FAQ*, <http://www.prosoniq.com/time_pitch_faq.html>, November 1996

Puckette 90 — **M. Puckette and D. Zicarelli**, *MAX – An Interactive Graphic Programming Environment*, Opcode Systems, 1990

Puckette 91a — **M. Puckette**, *FTS: A Real-Time Monitor for Multiprocessor Music Synthesis*, CMJ, 15(3), 56-67, Fall 1991

Puckette 91b — **M. Puckette**, *Combining Event and Signal Processing in the Max Graphical Programming Environment*, CMJ, 15(3), 68-77, Fall 1991

Quatieri 85 — **T.E. Quatieri and R.J. McAulay**, *Speech transformations based on a sinusoidal model*, Proc. ICASSP, March 1985

Quatieri 90 — **T.E. Quatieri and R.G. Danisewicz**, *An approach to co-channel talker interference suppression using a sinusoidal model for speech*, IEEE Trans. ASSP, 38(1), 1990

Rabiner 75 — **L.R. Rabiner and B. Gold**, *Theory and Application of Digital Signal Processing*, Englewood Cliffs: Prentice-Hall, 1975

Rabiner 78 — **L.R. Rabiner and R.W. Schafer**, *Digital Processing of Speech Signals*, Englewood Cliffs: Prentice-Hall, 1978

Radunskaya — **A. Radunskaya**, *Chaos and Non-linear Models*, Proc. ICMC, Hong Kong, 440-443, 1996

Railsback — **O.L. Railsback**, *Scale temperament as applied to piano tuning*, JASA, 9, 274, and 10, 86, 1938

Rakowski — **A. Rakowski**, *Pitch discrimination at the threshold of hearing*, Proc. of the Seventh International Congress on Acoustics, vol. 3, 1971

Rasch 78 — **R.A. Rasch**, *The Perception of Simultaneous Notes such as in Polyphonic Music*, Acustica, Vol. 40, 21-33, 1978

Rasch 82 — **R.A. Rasch and R. Plomp**, *The Perception of Musical Tones*, in The Psychology of Music, D. Deutsch, ed., 1-24, New York: Academic Press, 1982

Redding — **N.J. Redding and G.N. Newsam**, *Efficient Calculation of Finite Gabor Transforms*, IEEE Trans. Signal Processing, 44(2), 190-200, February 1996

Reekie — **H.J. Reekie and M. Meyer**, *The Host-Engine Software Architecture for Parallel Digital Signal Processing*, Proc. Australasian Workshop on Parallel and Real-Time Systems, Melbourne, 1994

Repp — **B.H. Repp**, *Diversity and commonality in music performance: An analysis of timing microstructure in Schumann's "Träumerei"*, JASA, 92(5), 2546-2568, 1992

Rioul 91 — **O. Rioul and M. Vetterli**, *Wavelets and Signal Processing*, IEEE Signal Processing Magazine, 14-38, October 1991

Rioul 92 — **O. Rioul and P. Duhamel**, *Fast Algorithms for Discrete and Continuous Wavelet Transforms*, IEEE Trans. Information Theory, 38(2), 569-586, 1992

Risset 69 — **J-C. Risset and M. Mathews**, *Analysis of Musical Instrument Tones*, Physics Today, 22(2), 23-30, 1969

Risset 82 **J-C. Risset and D.L. Wessel**, *Exploration of Timbre by Analysis and Resynthesis*, in The Psychology of Music, D. Deutsch, ed., 25-98, New York: Academic Press, 1982

Risset 91 **J-C. Risset**, *Paradoxical Sounds*, in Current Directions in Computer Music Research, M.V. Mathews and J.R. Pierce, eds., 149-158, Cambridge, Massachusetts: MIT Press, 1991

Roads 78 **C. Roads**, *Automated Granular Synthesis of Sound*, CMJ, 2(2), 61-62, Summer 1978

Roads 85 **C. Roads**, *Granular Synthesis of Sound*, Foundations of Computer Music, C. Roads and J. Strawn, eds., 145-159, Cambridge, Massachusetts: MIT Press, 1985

Roads 88 **C. Roads**, *Introduction to Granular Synthesis*, CMJ, 12(2), 11-13, Summer 1988

Roads 89 **C. Roads**, ed., *The Music Machine – Selected Readings from Computer Music Journal*, Cambridge, Massachusetts: MIT Press, 1989

Roads 91 **C. Roads,** *Asynchronous Granular Synthesis*, in G. De Poli, A. Piccialli, and C. Roads, eds., *Representations of Musical Signals*, Cambridge, Massachusetts: MIT Press, 1991

Roads 92 **C. Roads**, *Musical Applications of Advanced Signal Transformations*, Proc. Capri Workshop on Models and Representations of Musical Signals, Dept. of Physics, Univ. of Naples Federico II, 1992

Roads 93 **C. Roads**, *Musical Sound Transformation by Convolution*, Proc. ICMC, Tokyo, 102-109, 1993

Roads 94 **C. Roads**, *Computer Music Tutorial*, Cambridge, Massachusetts: MIT Press, 1994

RobinsonA **A.J. Robinson**, <ajr@softsound.com> *Shorten: simple lossless and near-lossless waveform compression*, <http://svr-www.eng.cam.ac.uk/~ajr/tr156/>, December 1994, and private email.

RobinsonK **K. Robinson and R.D. Patterson**, *The Duration Required To Identify the Instrument, the Octave, or the Pitch Chroma of a Musical Note*, Music Perception, 13(1), 1-15, Fall 1995

Rodet 92a **X.O. Rodet and Ph. Depalle**, *A new additive synthesis method using inverse Fourier transform and spectral envelopes*, Proc. ICMC, San Jose, 410-411, 1992

Rodet 92b **X.O. Rodet**, *Nonlinear Oscillator Models of Musical Instrument Excitation*, Proc. ICMC, San Jose, 412-413, 1992

Rodet 92c **X.O. Rodet and Ph. Depalle**, *Spectral Envelopes and Inverse FFT Synthesis*, Proc. AES Convention, 1992

Rodet 93 **X. Rodet**, *Flexible Yet Controllable Physical Models: A Nonlinear Dynamics Approach*, Proc. ICMC, Tokyo, 48-55, 1993

Rodet 96 **X. Rodet and C. Vergez**, *Physical Models of Trumpet-like Instruments: Detailed Behavior and Model Improvements*, Proc. ICMC, Hong Kong, 448-453, 1996

Ross **M.J. Ross, H.L. Shaffer, A. Cohen, R. Freudberg, and H.J. Manley**, *Average Magnitude Difference Function Pitch Extractor*, IEEE Trans. ASSP, 22, 353-362, 1974

Rossing — **T.D. Rossing**, *The Science of Sound*, Reading, Massachusetts: Addison-Wesley, 1990

Rossiter — **D. Rossiter and D.M. Howard**, *A graphical environment for electroacoustic music composition*, Proc. ICMC, Aarhus, 272-275, 1994

Roucous — **S. Roucous and A.M. Wilgus**, *High Quality Time-Scale Modification of Speech*, IEEE Trans. ASSP, 35(10), 1486-1487, 1987

Samson — **P.R. Samson**, *Architectural issues in the design of the system concepts digital synthesizer*, in Digital Audio Engineering: An anthology, J. Strawn, ed., Los Alamos: Kaufman, 1985

Sandell 91 — **G.J. Sandell**, *A Library of Orchestral Instrument Spectra*, Proc. ICMC, Montreal, 98-101, 1991

Sandell 95 — **G.J. Sandell**, *The SHARC Timbre Database*, <http://www.parmly.luc.edu/sharc/>, May 1995

Sano — **H. Sano and B.K. Jenkins**, *A Neural Network Model for Pitch Perception*, in Music and Connectionism, P.M. Todd and D.G. Loy, eds., Cambridge, Massachusetts: MIT Press, 1991

Sawada — **H. Sawada, N. Onoe, and S. Hashimoto**, *Acceleration Sensor as an Input Device for Musical Environment*, Proc. ICMC, Hong Kong, 421-424, 1996

Scallan — **C. Scallan and T. Stainsby**, *A New Software Package for Spectral Investigation and Analysis/Synthesis Using FFT and Sinusoidal Modelling Techniques*, Proc. ICMC, Tokyo, 399-401, 1993

Scavone — **G.P. Scavone**, *Modeling and Control of Performance Expression in Digital Waveguide Models of Woodwind Instruments*, Proc. ICMC, Hong Kong, 224-227, 1996

Scheirer 95a — **E.D. Scheirer**, *Extracting Expressive Performance from Recorded Music*, M.S. thesis, MIT Media Lab., 1995

Scheirer 95b — **E.D. Scheirer**, *Using Musical Knowledge to Extract Expressive Performance Information from Audio Recordings*, Proc. Int. Joint Conf. on Artificial Intelligence, Workshop on Computational Auditory Scene Analysis, Montreal, August 1995, and <ftp://sound.media.mit.edu/pub/Papers/eds-ijcai95.ps.gz>

Scheirer 96 — **E.D. Scheirer**, *Some thoughts on the transcription problem – or, why aren't there good audio-to-MIDI converters available?*, <http://sound.media.mit.edu/~eds/transcription.html>, 13 January 1996

Schloss — **W.A. Schloss**, *On the Automatic Transcription of Percussive Music – from Acoustics Signal to High-Level Analysis*, PhD thesis, and Stanford Music Dept. Technical Report STAN-M-27, 1985

Schottstaedt — **W. Schottstaedt**, *An Introduction to FM*, <http://www.notam.uio.no/~andersvi/home/cm-sys/clm/fm.html>, 1991

Schroeder — **M.R. Schroeder**, *Period Histogram and Product Spectrum: New Methods for Fundamental-Frequency Extraction*, JASA, 829-834, 1968

Schuck — **O.H. Schuck and R.W. Young**, *Observations on the Vibrations of Piano Strings*, JASA, 15, 1-11, 1943, reprinted in Musical Acoustics – Piano and Wind Instruments, E.L. Kent, ed., Stroudsburg: Dowden, Hutchinson and Ross, 1977

SciTech — **SciTech Software**, *UniVesa documentation*, <http://www.scitechsoft.com/>, 1995

Serra 89 — **X. Serra**, *A system for sound analysis/transformation/synthesis based on a deterministic plus stochastic decomposition*, PhD thesis, Stanford Univ., October 1989

Serra 90 — **X. Serra and J.O. Smith**, *Spectral Modeling Synthesis: A Sound Analysis/Synthesis System Based on a Deterministic plus Stochastic Decomposition*, CMJ, 14(4), 12-24, Winter 1990

Serra 94 — **X. Serra**, *Sound hybridization based on a deterministic plus stochastic decomposition model*, Proc. ICMC, Aarhus, 348-351, 1994

Serra 96 — **X. Serra**, *Musical Sound Modeling With Sinusoids Plus Noise*, <http://www.iua.upf.es/eng/recerca/mit/sms/articles/msm/>, 1996

Settel — **Z. Settel and C. Lippe**, *Musical Applications Using Real-time Frequency Domain Signal Processing*, Proc. XI Colloquium on Musical Informatics, Bologna, 13-17, 1995

Shah — **I.A. Shah and A.A.C. Kalker**, *Theory and design of multidimensional QMF sub-band filters from 1-D filters and polynomials using transforms*, IEE Proceedings, 140(1), 67-71, 1993

Shankland — **R.S. Shankland and J.W. Coltman,** *Overtones of a vibrating string*, in Musical Acoustics – Piano and Wind Instruments, E.L. Kent, ed., Dowden, Hutchinson and Ross, Stroudsburg, Pennsylvania, 1977

Shephard — **R.N. Shephard**, *Structural Representations of Musical Pitch*, in The Psychology of Music, D. Deutsch, ed., Academic Press, New York, 343-390, 1982

Shower — **E.G. Shower and R. Biddulph**, *Differential Pitch Sensitivity of the Ear*, JASA, 3, 274, 1931

Shuttleworth — **T. Shuttleworth and R.G. Wilson**, *Note Recognition in Polyphonic Music using Neural Networks*, Dept. of Comp. Sci. Technical Report CS-RR-252, October 1993, and <ftp://ftp.dcs.warwick.ac.uk/pub/reports/rr/ 252/>

Silberg — **S. Silberg**, *Intel i860 versus Digital Signal Processors (DSP)*, Microprocessing and Microprogramming, 35, 605-610, 1992

Slaney 95a — **M. Slaney, D. Ellis, and D. Rosenthal**, *Report on the Computational Auditory Scene Analysis Workshop*, <http://sound.media.mit.edu/~dfr/casa/summary.html>, 1995

Slaney 95b — **M. Slaney**, *A Critique of Pure Audition*, Proc. Computational Auditory Scene Analysis Workshop, Int. Joint Conf. on Artificial Intelligence, Montreal, August 1995

Slawson — **A.W. Slawson**, *Vowel quality and musical timbre as functions of spectrum envelope and fundamental frequency*, JASA, 43, 87-101, 1968

SmithJ 87 — **J.O. Smith and X. Serra**, *PARSHL: An analysis/synthesis program for non-harmonic sounds based on a sinusoidal representation*, Proc. ICMC, San Francisco, 290-297, 1987

SmithJ 91 — **J.O. Smith**, *Viewpoints on the History of Digital Synthesis*, Proc. ICMC, Montreal, 1-10, 1991

SmithJ 92 — **J.O. Smith**, *Physical Modeling Using Digital Waveguides*, CMJ, 16(4), 74-91, Winter 1992

SmithJ 93 **J.O. Smith**, *Efficient Synthesis of Stringed Musical Instruments,* Proc. ICMC, Tokyo, 64-71, 1993

SmithM 84 **M.J.T. Smith and T.P. Barnwell**, *A procedure for designing exact reconstruction filterbanks for tree-structured coders,* Proc. IEEE ICASSP, 27.1-4, San Diego, March 1984

SmithM 85 **M.J.T. Smith and T.P. Barnwell**, *A unifying framework for analysis/synthesis systems based on maximally decimated filter bands,* Proc. IEEE ICASSP, 521-524, Tampa, March 1985

SmithM 86 **M.J.T. Smith and T.P. Barnwell**, *Exact Reconstruction Techniques for Tree-Structured Subband Coders,* IEEE Trans. ASSP, 34(3), 434-441, 1986

Smyth **S.M.F. Smyth and J.V. McCanny**, *High-fidelity Music Coding at 4 bits,* Electronic Letters, 24(8), 493-495, 1988

Snell **J. Snell**, *Design of a Digital Oscillator That Will Generate up to 256 Low-Distortion Sine Waves in Real Time,* CMJ, 1(2), 4-25, Summer 1977

Solbach 96a **L. Solbach and R. Wöhrmann**, *Sound Onset Localization and Partial Tracking in Gaussian White Noise,* Proc. ICMC, Hong Kong, 324-327, 1996

Solbach 96b **L. Solbach, R. Wöhrmann, and J. Kliewer**, *The complex-valued continuous wavelet transform as a preprocessor for auditory scene analysis,* in H. Okuno and D. Rosenthal, eds., Readings in Computational Auditory Scene Analysis, Hillsdale: Erlbaum, 1996, and <ftp://ftp.ti6.tu-harburg.de/pub/paper/ijcai95-casa_rev1.ps.gz>, 30 January 1996

Sondhi **M.M. Sondhi**, *New Methods of Pitch Extraction,* IEEE Trans. Audio Electroacoustics, AU-16, 262-268, 1968

Srinivasan **R. Srinivasan**, *Auditory Critical Bandwidth for Short-Duration Signals,* JASA, 50(2), 616-622, 1971

Stainsby **T. Stainsby**, *A System for the Separation of Simultaneous Musical Audio Signals,* Proc. ICMC, Hong Kong, 75-78, 1996

Sterian **A. Sterian and G.H. Wakefield**, *Robust Automated Music Transcription Systems,* Proc. ICMC, Hong Kong, 219-221, 1996

Stevens **S.S. Stevens**, *The Relation of Pitch to Intensity,* JASA, 6, 150, 1935

Stockham 69 **T.G. Stockham**, *High-speed convolution and correlation with applications to digital filtering,* in Digital Processing of Signals, B. Gold and C.M. Rader, eds., New York: McGraw-Hill, 1969

Stockham 75 **T.G. Stockham, T.M. Cannon, and R.B. Ingerbretsen**, *Blind deconvolution through digital signal processing,* Proc. IEEE, 63(4), 678-692, April 1975

Strawn **J. Strawn**, *Approximation and Syntactic Analysis of Amplitude and Frequency Functions for Digital Sound Synthesis,* CMJ, 4(3), 3-24, Fall 1980

Sundberg 73 **J.E.F. Sundberg and J. Lindqvist**, *Musical octaves and pitch,* JASA, 54(4), 922-929, 1973

Sundberg 91 **J.E.F. Sundberg**, *The Science of Musical Sounds,* San Diego: Academic Press, 1991

Swaminathan **K. Swaminathan and P.P. Vaidyanathan**, *Theory and Design of Uniform DFT, Parallel, Quadrature Mirror Filters,* IEEE Trans. on Circuits and Systems, 33(12), 1170-1191, December 1986

Sweldens 93 **W. Sweldens and R. Piessens**, *Wavelet sampling techniques,* Proc. of the Statistical Computing Section, American Statistical Association, 20-29, 1993

Sweldens 96 **W. Sweldens and P. Schröder**, *Building Your Own Wavelets at Home,* in Wavelets in Computer Graphics, ACM SIGGRAPH Course Notes, 1996

Syntrillium **Syntrillium Software Corp.** <syntrill@aol.com>, *Cool Edit,* <http://www.syntrillium.com/cool.htm>, November 1996

Szilas **N. Szilas**, *Physical Models That Learn,* Proc. ICMC, Tokyo, 72-75, 1993

Tait **C. Tait,** *Audio Analysis for Rhythmic Structure*, Proc. ICMC, Banff, 590-591, 1995

Takami **K. Takami, H. Katayose, and S. Inokuchi**, *Extraction of Performance Information in Piano Music*, Trans. Institute of Electronics, Information and Communication Engineers, 1989 [in Japanese]

Takeuchi **N. Takeuchi, H. Katayose, and S. Inokuchi,** *Virtual Performer: Adaptive KARAOKE system*, Convention Record of Information Processing Society of Japan, Spring 1993 [in Japanese]

Tanguiane 87 **A.S. Tanguiane**, *Raspoznavanie akkordov pri avtomaticeskoi notnoi transkripcii polifoniceskoi muzyki,* Sojuz Kompozitorov SSSR / Akademia Nauk SSSR, Moscow, 1987 [in Russian]

Tanguiane 88 **A.S. Tanguiane**, *An Algorithm for Recognition of Chords,* Proc. ICMC, Cologne, 199-210, 1988

Tanguiane 91 **A.S. Tanguiane**, *Criterion of Data Complexity in Rhythm Recognition,* Proc. ICMC, Montreal, 559-562, 1991

Tanguiane 93a **A.S. Tanguiane**, *Artificial Perception and Music Recognition,* Springer-Verlag, Berlin, 1993

Tanguiane 93b **A.S. Tanguiane**, *An Artificial Perception Model and Its Application to Music Recognition,* Proc. ICMC, Tokyo, 284-291, 1993

Tanguiane 95 **A.S. Tanguiane**, *Towards Axiomatization of Music Perception,* JNMR, 24, 247-281, 1995

Taylor 94 **I.J. Taylor and M. Greenhough**, *Evaluation of artificial-neural-network Pitch types for the determination of pitch,* Proc. ICMC, Aarhus, 114-120, 1994

Taylor 95 **I.J. Taylor and M. Greenhough**, *Neural Network Pitch Tracking over the Pitch Continuum,* Proc. ICMC, Banff, 432-435, 1995

Technics **Technics**, *SC-CH550 Operating Instructions,* Matsushita Electric Industrial Co. Ltd.

Terhardt 71 **E. Terhardt**, *Die Tonhöhe harmonischer Klänge und das Oktavintervall,* Acustica, 24, 126-136, 1971

Terhardt 78 **E. Terhardt**, *Psychoacoustical evaluation of musical sounds,* Perception and Psychophysics, 23, 483-492, 1978

Terhardt 79 **E. Terhardt**, *Calculating Virtual Pitch,* J. Hearing Research, 1, 155, 1979

Terhardt 82a **E. Terhardt, G. Stoll, and M. Seewann**, *Algorithm for extraction of pitch and pitch salience from complex tonal signals,* JASA, 71(3), 679-688, 1982

Terhardt 82b — **E. Terhardt, G. Stoll, and M. Seewann**, *Pitch of complex signals according to virtual-pitch theory: Tests, examples, and predictions*, JASA, 71(3), 671-678, 1982

Texas — **Texas Instruments Inc.**, *TMS320C4x User Guide*, Literature number SPRU063, 1991

Therrien — **C.W. Therrien, R. Cristi, and O.E. Kjono**, *Analysis/Synthesis of Sound Using a Time-Varying Linear Model*, Proc. ICMC, Aarhus, 331-332, 1994

ThreeL — **3L Ltd.**, *Parallel C User Guide*, 3L Ltd., Edinburgh, 1992

Todd — **N.P.McA. Todd**, *Wavelet Analysis of Rhythm in Expressive Musical Performance*, Proc. ICMC, Tokyo, 264-267, 1993

Todoroff — **T. Todoroff, E. Daubresse, and J. Fineberg**, *Iana – a real-time environment for analysis and extraction of frequency components of complex orchestral sounds and its application within a musical realization*, Proc. ICMC, Banff, 292-293, 1995

Toiviainen — **P. Toiviainen**, *Optimizing Self-Organizing Timbre Maps: The Effect of Auditory Images and Distance Metrics*, Proc. XI Colloquium on Musical Informatics, Bologna, 141-144, 1995

Transtech — **Transtech Ltd.**, *TDMB416 User Guide*, Transtech Ltd., High Wycombe, Buckinghamshire

Truax 87 — **B. Truax**, *Real-Time Granulation of Sampled Sound with the DMX-1000*, Proc. ICMC, Illinois, 1987

Truax 88 — **B. Truax**, *Real-Time Granular Synthesis with a Digital Signal Processor*, CMJ, 12(2), 14-26, Summer 1988

Truax 90 — **B. Truax**, *Time Shifting of Sampled Sound with a Real-Time Granulation Technique*, Proc. ICMC, Glasgow, 104-107, 1990

Truax 91 — **B. Truax**, *Composition with time-shifted environmental sound using a real-time granulation technique*, Proc. ICMC, Montreal, 487-490, 1991

Truax 93 — **B. Truax**, *Time Shifting and Transposition of Sampled Sound with a Real-Time Granulation Technique*, Proc. ICMC, Tokyo, 82-85, 1993

Truax 94 — **B. Truax**, *Discovering Inner Complexity: Time Shifting and Transposition with a Real-time Granulation Technique*, CMJ, 18(2), 38-48, Summer 1994

Tsujimoto — **K. Tsujimoto, M. Imai, and S. Inokuchi**, *Assistance Tool for Ethnic Music Recognition*, Information Processing Society of Japan, 1986 [in Japanese]

Ueda — **M. Ueda and S. Hashimoto,** *Blind Decomposition of Concurrent Sounds*, Proc. ICMC, Aarhus, 311-318, 1994

Ungvary — **T. Ungvary and S. Waters,** *The sonogram: a tool for visual documentation of musical structure*, Proc. ICMC, Glasgow, 159-162, 1990

Vaidyanathan 87 — **P.P. Vaidyanathan,** *Theory and Design of M-Channel Maximally Decimated Quadrature Mirror Filters with Arbitrary M, Having the Perfect-Reconstruction Property*, IEEE Trans. ASSP, 35(3), 1987

Vaidyanathan 90 — **P.P. Vaidyanathan,** *Multirate digital filters, filter banks, polyphase networks, and applications: a tutorial*, Proc. IEEE, 78(1), 56-93, 1990

Välimäki 93 — **V. Välimäki, M. Karjalainen, and T.I. Laakso,** *Modeling of Woodwind Bores with Finger Holes,* Proc. ICMC, Tokyo, 32-39, 1993

Välimäki 96 — **V. Välimäki, R. Hänninen, and M. Karjalainen,** *An Improved Digital Waveguide Model of a Flute – Implementation Issues,* Proc. ICMC, Hong Kong, 1-4, 1996

Van Duyne 93 — **S.A. Van Duyne and J.O. Smith,** *Physical Modeling with the 2-D Digital Waveguide Mesh,* Proc. ICMC, Tokyo, 40-47, 1993

Van Duyne 96 — **S.A. Van Duyne and J.O. Smith,** *The 3D Tetrahedral Digital Waveguide Mesh with Musical Applications,* Proc. ICMC, Hong Kong, 9-16, 1996

Van Klitzing — **R. Van Klitzing and A. Kohlrausch,** *Effect of masker level on overshoot in running- and frozen-noise maskers,* JASA, 95(4), 2192-2201, 1994

Vercoe 84 — **B. Vercoe,** *The Synthetic Performer in the context of Live Performance*, Proc. ICMC, Paris, 199-200, 1984

Vercoe 85 — **B. Vercoe and M. Puckette**, *Synthetic Rehearsal: Training the Synthetic Performer*, Proc. ICMC, Vancouver, 275-278, 1985

Vercoe 88 — **B. Vercoe and D. Cumming**, *Connectionist Machine Tracking of Polyphonic Audio*, Proc. ICMC, Cologne, 211-218, 1988

Vercoe 90 — **B. Vercoe and D.P.W. Ellis**, *Real-time Csound: Software Synthesis with Sensing and Control*, Proc. ICMC, Glasgow, 209-211, 1990

Vercoe 93 — **B. Vercoe**, *Csound: A Manual for the Audio Processing System and Supporting Programs with Tutorials*, Media Laboratory, MIT, 1993

Vercoe 96 — **B. Vercoe**, *Extended Csound*, Proc. ICMC, Hong Kong, 141-142, 1996

Verge — **M.-P. Verge,** *Physical modeling of aeroacoustic sources in flute-like musical instruments,* Proc. ICMC, Hong Kong, 5-8, 1996

von Békésy — **G. von Békésy,** *Experiments in Hearing,* New York: McGraw-Hill, 1960

von Bismarck — **G. von Bismarck,** *Timbre of steady sounds: A factorial investigation of its verbal attributes,* Acustica, 30, 146-159, 1974

Wake 92 — **S. Wake, H. Kato, N. Saiwaki, and S. Inokuchi,** *The Session System Reacting to the Sentiment of the Player*, Japan Music and Computer Science Society, Proc. of Summer Symposium, 1992 [in Japanese]

Wake 94 — **S. Wake, H. Kato, N. Saiwaki, and S. Inokuchi**, *Cooperative musical partner system using tension parameter: JASPER (jam session partner)*, Trans. IPS Japan, 35(7), 1469-1481, 1994 [in Japanese]

Wallraff — **D. Wallraff**, *The DMX-1000 Signal Processing Computer*, CMJ, 3(4), 44-49, Winter 1979

Wang — **A. Wang**, *Instantaneous and Frequency-Warped Signal Processing Techniques for Auditory Source Separation*, PhD thesis, Dept. of Elec. Engg., Stanford Univ., Stanford, California, August 1994

Ward — **W.D. Ward and E.M. Burns**, *Absolute Pitch*, in The Psychology of Music, D. Deutsch, ed., New York: Academic Press, 431-451, 1982

Watson — **C. Watson**, *The Computer Analysis of Polyphonic Music*, PhD thesis, Sydney Univ., 1986

Wawrzynek 84 — **J. Wawrzynek, C. Mead, L. Tzu-mu, and L. Dyer**, *A VLSI approach to sound synthesis*, Proc. ICMC, Paris, 53-64, 1984

Wawrzynek 91 — **J. Wawrzynek**, *VLSI Models for Sound Synthesis*, in Current Directions in Computer Music Research, M.V. Mathews and J.R. Pierce, eds., 185-198, Cambridge, Massachusetts: MIT Press, 1991

Weintraub — **M. Weintraub,** *A Theory and Computational Model of Auditory Monaural Sound Separation,* PhD thesis, Dept. of Elec. Eng., Stanford Univ., 1985

Weiss — **L.G. Weiss,** *Wavelets and Wideband Correlation Processing,* IEEE Signal Processing Magazine, 13-32, January 1994

Wessel 78 — **D.L. Wessel,** *Low dimensional control of musical timbre,* Proc. 59th Conv. AES, Hamburg, 1978

Wessel 79 — **D.L. Wessel,** *Timbre space as a musical control structure,* CMJ, 3(2), 45-52, Summer 1979

Wickerhauser — **M.V. Wickerhauser,** *Acoustic Signal Compression with Wave Packets,* <http://www.math.yale.edu/pub/wavelets/papers/acoustic.tex>, 1989

Widmer — **G. Widmer,** *Learning Expressive Performance: The Structure-Level Approach,* JNMR, 25(2), 179-205, June 1996

Wiggins — **G. Wiggins, E. Miranda, A. Smaill, and M. Harris,** *A Framework for the Evaluation of Music Representation Systems,* CMJ, 17(3), 31-42, Fall 1993

Wildcat — **Wildcat Canyon Software,** *Autoscore,* <http://www.wildcat.com/Pages/Autoscor.htm>, 1996

Williamson — **R. Williamson** <76570.2752@compuserve.com>, *Video for DOS,* <gopher://micros.hensa.ac.uk:70/11/micros/ibmpc/dos/l/l095/>, January 1995

Wilson 92a — **R.G. Wilson, A.D. Calway, and E.R.S. Pearson,** *A Generalized Wavelet Transform for Fourier Analysis: The Multiresolution Fourier Transform and Its Application to Image and Audio Signal Analysis,* IEEE Trans. Information Theory, 38(2), 674-690, March 1992

Wilson 92b — **R.G. Wilson, A.D. Calway, E.R.S. Pearson, and A.R. Davies,** *An Introduction to the Multiresolution Fourier Transform and Its Applications,* Warwick Univ. Dept. of Comp. Sci. Technical Report 204, January 1992, and <ftp://ftp.dcs.warwick.ac.uk/pub/reports/rr/204/>

Winckel — **F. Winckel,** *Measurements of the Acoustic Effectiveness and Quality of Trained Singers' Voices,* Proc. 90th meeting Acoust. Soc. of America, San Francisco, 1975

Wöhrmann — **R. Wöhrmann and L. Solbach,** *Preprocessing for the Automated Transcription of Polyphonic Music: Linking Wavelet Theory and Auditory Filtering,* Proc. ICMC, Banff, 396-399, 1995

Wood — **A. Wood,** *The Physics of Music,* London: Chapman and Hall, 1976

Xenakis — **I. Xenakis**, *Formalized Music,* Bloomington: Indiana University Press, 1971 (and Pendragon, 1991)

Yamaha — **Yamaha Corporation**, *Yamaha SY77 Operating Manual,* Hamamatsu, Japan

Zwicker — **E. Zwicker, G. Flottorp, and S.S. Stevens**, *Critical Bandwidth in Loudness Summation,* JASA, 29, 548, 1957

Internet references are cited according to M. Page <pagem@etsuarts.east-tenn-st.edu>, *A brief citation guide for Internet sources in History and the Humanities*, version 2.1, <http://h-net.msu.edu/~africa/citation.html>, February 1996. It is noted that World-Wide Web references may move, disappear, or be updated.

This thesis will be made available via <http://capella.dur.ac.uk/doug/thesis/>. An updated list of the web links in the references, and the audio examples, can also be found via this page.

11.2 Abbreviations

AES	Audio Engineering Society
ASSP	Acoustics, Speech, and Signal Processing
CCRMA	Center for Computer Research in Music and Acoustics
CMJ	Computer Music Journal
ICMC	International Computer Music Conference
ICSPAT	International Conference on Signal Processing Applications and Technology
IEE	Institute of Electrical Engineers
IEEE	Institute of Electrical and Electronic Engineering
ICASSP	International Conference on Acoustics, Speech, and Signal Processing
IRCAM	Institut de Recherche et Coordination Acoustique/Musique
JASA	Journal of the Acoustical Society of America
JNMR	Journal of New Music Research
MIT	Massachusetts Institute of Technology
SIAM	Society for Industrial and Applied Mathematics
WASPAA	Workshop on Applications of Signal Processing to Audio and Acoustics

11.3 Translations

Translations of foreign titles are as follows:-

Auger — *La réallocation: une méthode générale d'amélioration de la lisibilité des représentation temps-fréquence bilinéaires*
Reallocation: a general method to improve the readability of bilinear time-frequency representations

Bürck — *Die Hörbarkeit von Laufzeitdifferenzen*
The audibility of propagation-time differences

Heinbach 87 — *Gehörgerecht Repräsentation von Audiosignalen durch das Teiltonzeitmuster*
Aurally accurate representations of audio signals using partial envelopes

Höldrich 94a — *Zur Analyse und Resynthese von Klangsignalen unter Verwendung von Zeit-Frequenze-Repräsentationen mit Verbesserter Lokalisation der Signalenergie*
On the analysis and resynthesis of sound signals using a time-frequency representation with improved localisation of signal energy

Kodera 76 — *Analyse numérique de signeaux géophysiques nonstationaires*
Numerical analysis of nonstationary geophysical signals

Tanguiane 87 — *Raspoznavanie akkordov pri avtomaticeskoi notnoi transkripcii polifoniceskoi muzyki*
Perception of chords in automatic note transcription of polyphonic music

Terhardt 71 — *Die Tonhöhe harmonischer Klänge und das Oktavintervall*
Tone height of harmonic sounds and the octave interval

11.4 Company addresses

American Paper Optics	American Paper Optics, 2005 Nonconnah Boulevard, Suite 27, Memphis, TN 38132, USA
Atlanta	Atlanta Signal Processors Inc., 1375 Peachtree St. NE, Suite 690, Atlanta, GA 30309-3115, USA
Borland	Borland International Inc., 1800 Green Hills Road, PO Box 660001, Scotts Valley, CA 95066-0001, USA
Emagic	Emagic Inc., 13348 Grass Valley Ave., Bldg C, Suite 100, Grass Valley, CA 95945, USA
Gravis	Advanced Gravis Computer Technology Ltd., 101-3750 North Fraser Way, Burnaby, B.C. V5J 5E9, Canada
Hohner	Hohner Midia, Schwabbenstraße 27, D-74626 Bretzfeld, Germany
Jordan Hargraphics	Jordan Hargraphics Software, 8760-A Research Boulevard #256, Austin, TX 78758, USA
LT Sound	LT Sound Inc., 7980 LT Parkway, Lithonia, GA 30058, USA
Matsushita	Matsushita Electric Industrial Co. Ltd., Central P.O. Box 288, Osaka 530-91, Japan
Microsoft	Microsoft Inc., 1 Microsoft Way, Redmond, WA 98052-6399, USA
Opcode	Opcode Systems Inc., 3950 Fabian Way, Suite 100, Palo Alto, CA 94303, USA
Pkware	Pkware Inc., 91025 North Deerwood Drive, Brown Deer, WI 53223, USA
SciTech	SciTech Software Inc., 5 Governors Lane, Suite D, Chico, CA 95926-1989, USA
Syntrillium	Syntrillium Software Corp., P.O. Box 60274, Phoenix, AZ 85082-0274, USA
Texas Instruments	Texas Instruments Inc., 12501 Research Boulevard, Austin, TX 78759, USA
3L	3L Ltd., 86/92 Causewayside, Edinburgh EH1 1PY, UK
Transtech	Transtech Ltd., 17-19 Manor Court Yard, Hughenden Avenue, High Wycombe, Buckinghamshire, HP13 5RE, UK
Wildcat	Wildcat Canyon Software, 1563 Solano Avenue #264, Berkeley, CA 94707, USA
Yamaha	Yamaha Corporation, Nakazawa-cho 10-1, Hamamatsu, Japan 430

11.5 Papers presented during the course of this research

Nunn 94 — **D.J.E. Nunn, A. Purvis, and P.D. Manning**, *Source Separation and Transcription of Polyphonic Music*, Proc. International Colloquium on New Music Research, Gënt, Belgium, 1994

Nunn 95 — **D.J.E. Nunn, A. Purvis, and P.D. Manning**, *Graphical display of musical information*, Proc. XI Colloquium on Musical Informatics, Bologna, 235-236, 1995

Itagaki 95b — **T. Itagaki, D.J.E Nunn, D.K. Phillips, D. Batjakis, A. Purvis, and P.D. Manning**, *Activity Report*, XI Colloquium on Musical Informatics, Bologna, 51-54, 1995

Nunn 96 — **D.J.E. Nunn, A. Purvis, and P.D. Manning**, *Acoustic Quanta*, Proc. ICMC, Hong Kong, 52-54, 1996

Itagaki 96b — **T. Itagaki, S. Johnson, P.D. Manning, D.J.E. Nunn, D.K. Phillips, A. Purvis, and J. Spanier**, *Durham Music Technology: Activity Report*, Proc. ICMC, Hong Kong, 126-128, 1996

12. Acknowledgements

First I thank my parents Margaret and Clifford Nunn for their love, support, and encouragement during this research, and before, and beyond.

I would like to express my sincere thanks to my supervisor Professor Alan Purvis, for his valuable guidance on the principles of DSP and for his constant encouragement and practical advice. Thanks are also due to Dr Peter Manning for sharing his extensive musical expertise. It is also a pleasure to acknowledge the contributions made by my friends and colleagues: Takebumi Itagaki, for many useful discussions regarding the transputer architecture and the problems of synthesis; Des Phillips, for sharing his in-depth knowledge of computer architecture; Matthew Jubb, for his moral support and UNIX wizardry; Jonathan Spanier, for his systems programming expertise; and Dionissios Batjakis for stimulating discussions. I would also like to thank Milos Kolar for patiently designing and programming the output board for the C40, and Peter Friend and Trevor Nancarrow for keeping things running as smoothly as possible. The staff of the IT service provided dozens of small but vital answers over the years. Countless and nameless others in the engineering and music departments are also appreciated for their valuable suggestions.

Texas Instruments are to be applauded for releasing their DSP assembler source code into the public domain. Transtech and 3L both provided much helpful support in the implementation of the C40 setup in the laboratory. Thanks must also go to American Paper Optics for providing the anaglyphic stereo glasses.

Tony Robinson of SoftSound Ltd. and various anonymous reviewers of papers gave many helpful comments. Many others in the wider research community that I met virtually via email and the usenet newsgroups comp.music.research, comp.dsp, alt.sci.physics.acoustics, and bionet.audiology are also gratefully acknowledged. Closer to home, many friends in the Graduate Society including Iain May, Dean Wood, Simon Brown, and Stefan Calvert helped me retain a degree of sanity over the course of this research.

This work has been funded by the Engineering and Physical Sciences Research Council.