MIDI, Digital Audio, and Software Synthesis, as Employed in the Composition of *The Triangle Suite*

Based on The Triangle Shirtwaist Fire of 1911

Master's Thesis
Lee Zakian
NYU – Music Technology
May 9th, 2002

Table of Contents

Preface P. 2

Introduction P. 7

I. MIDI P. 9

II. Digital Audio P. 16

III. Software Synthesis P. 22

IV. The Making of The Triangle Suite P. 38

Conclusion P. 45

Appendix 1
Glossary P. 46

Appendix 2
Examples P. 47

Appendix 3
Sound Effects P. 48
Mvt. I, II, III, IV, VI

Appendix 4
Csound Orchestras & Scores P. 49
Mvt. V, Mvt. VI Space & Vocal

Footnotes P. 71

Bibliography P. 73

Preface

I became interested in music technology about 12 years ago, as a woodwind performer and teacher who wanted to explore new avenues of sound. I purchased a Korg M-1 keyboard, which is still active in my studio, and I soon began to develop an understanding of the many parameters which make up a complete sound. Through the reshaping of envelopes and the interchanging of oscillators with different waveforms, I was able to program my own banks of MIDI sounds, which were often more pleasing to me than the factory presets which came loaded with the synthesizer.

I used the onboard sequencer to input accompaniments with which to perform. At first, I used these accompaniments as a teaching aid for my private students. Later, as my efforts became more sophisticated, I started giving solo performances with my portable orchestra. I was able to work as a solo act, but with the added interest of an orchestral accompaniment. I was still unaware of the inner workings of the MIDI protocol, and as yet I had almost no knowledge of the processes of converting sound to and from the digital domain. I was just thankful that the technology had been invented which allowed me to manipulate the tempo and the duration of musical notes without changing their pitch!

In 1993, I started an online publishing company, JL Publishing. Although remaining quite small, I have used it to distribute my MIDI accompaniments for almost a decade. Most of the publications are pieces which I currently perform, including the baroque, classical, and romantic orchestrations I used for a recital at NYU in November of 2000. The website I created on the Internet was a natural outgrowth of my efforts, as I moved from hardware synthesizer editing to software synthesizer editing. I found it to be so much more practical to create MIDI sequences with software programs, which could then be played through or loaded into a hardware synthesizer.

In the early 1990's I started using some rather primitive digital editing software, which opened up a new aspect of music technology to me. Unlike my MIDI endeavors, I was now actually dealing with sound itself. The speed of my 386 66 MHz computer with 4 Mg of RAM and a 40 Mg hard drive did not exactly offer me a satisfying introduction into this field, but I did get the chance to see and hear how precision edits could be made without the time consuming and difficult art of splicing, and how effects and processing could be applied to audio files, without permanently adding them to a tape.

Editing was mostly destructive at this time, with only a few programs offering non-destructive editing through Edit Decision Lists (EDL's). This resulted in unbearably long processing times, often taking hours to equalize a five minute song. MIDI and digital audio were still separate entities in most software programs at the time, but soon they became integrated into every level of consumer software. Today, most audio programs, and all computers, offer MIDI and digital audio.

My interest grew, but I found myself limited in my understanding of the inner workings of digital sound. I was adequate at using software, but I had no idea what was going on under the hood. I realized that I needed some formal education to obtain the level of knowledge I desired. I read information about several degree programs given throughout the country, but none had more to offer than the one at NYU. As a teacher and a performer in the Metropolitan area, my only realistic choice was to find a program within commuting distance from my home in New Jersey. Fortunately, NYU's Music Technology program happened to be the best one I could find, as well as the most convenient one for my situation.

My three years working toward a Master's Degree in Music Technology at NYU have been both enjoyable and enlightening. The faculty has been helpful as well as informative, and rarely was I unable to find a quick solution when a problem arose. There were times when the equipment didn't perform as expected, but those were some of the most beneficial, (and frustrating) moments I spent in the studios. Troubleshooting technical problems brought me to a better and a deeper understanding of music technology than I might have garnered if everything had always worked perfectly. Whether it was because of my own errors, or the fault of a particular piece of hardware or software, I often gained more practical experience from my failures than from my successes. I now feel confident in my understanding of digital technology, although there will always be more to learn. I plan to continue studying music technology, as new innovations become part of its ever-growing landscape.

I would like to thank the many fine professors who have guided me through this journey, including Dr. Kenneth Peacock, Dr. Robert Rowe, Dr. Richard Boulanger, Professor Sean Huff, Professor Barry Greenhut, and Professor Kevin Larke. Their knowledge and their teaching styles were quite different from each other, which made the program always interesting and challenging to me. I would especially like to thank Dr. Boulanger, who introduced me to Csound, a program I am just beginning to understand, but one I expect to be a part of me for the rest of my life.

I had my first inspiration for this project about two years ago, while watching a documentary film about the Triangle Shirtwaist Fire of 1911. I had heard about this disaster some years earlier, but I was unaware of its connection with NYU. I didn't realize that the building involved was now a part of NYU, or that the NYU Law students from the adjacent building were instrumental in saving many of the survivors on that Saturday afternoon. These revelations caused me to research the historical records of the event, by reading several books and newspaper accounts, and I decided I would like to compose a piece of programmatic music for my Master's Thesis, employing several disciplines of music technology.

For those people unaware of the situation which occurred, an overcrowded and inadequately safeguarded factory was responsible for the deaths of 146 young female garment workers on the 25th of March, 1911. The victims who did not perish inside, jumped to their deaths, eight or nine stories below. They landed on the pavement in front of the Asch Building, located at the corner of Washington Street and Greene Street. (The building now houses NYU's chemistry classes.) Many people were saved by climbing over to the NYU Law School building, aided by a ladder which was lowered to the Asch Building by students attending classes on that day. The victims were mostly Jewish and Italian immigrants, between the ages of thirteen and twenty-three. A number of factors contributed to the magnitude of the disaster: locked exit doors, flammable fabric, illegal smoking, inadequate stairways and elevators, and no sprinkler system. Ironically, the building itself was fireproof, and that is why it is still in use to this day. The death of these woman, in about 20 minutes time, led to sweeping reforms of the safety regulations required in factories. Groups, such as *The International Ladies' Garment Workers' Union*, were instrumental in seeing that incidents like this were avoided in the future.

As might be imagined, the subject matter of this composition is painfully close to the tragedy which befell our city and our nation on September 11th, 2001. This posed a more difficult problem for me than any musical or technical consideration could present. Due to the current sensitivity of New York City regarding the recent disaster, I have chosen to down-play the programmatic elements of my composition as much as possible, while emphasizing the musical and the technical aspects. Graphic footage, showing pictures and video of the event, have been eliminated, leaving interpretation to the individual imaginations of the audience. In this way, I hope that no one will be offended by the content of this piece, whose conception predated the events of September 11th by well over a year.

The body of this paper will focus on Csound and software synthesis, following chapters on MIDI and Digital Audio. Csound was used to create and process many of the sounds used in the Suite, as well as to render one entire movement. MIDI was used to create the other orchestral instruments, with different ensembles being employed in each movement. Digital audio was used to add sound effects where Csound was not used, as well as to edit and orchestrate the scores when they were imported into Pro Tools. These diverse elements will be examined in relation to their presence in the movements of the composition. At times these methods of music technology are used exclusively from each other, at times they are used simultaneously, and at times they used in conjunction with a live solo flute part. The overall objective of the composition is to come across as a chronological portrayal of the events on March 25th, 1911, without having the diverse technical elements detract from the flow of the music.

MIDI, or *Musical Instrument Digital Interface*, is a communication protocol which was introduced to the commercial marketplace in 1982. Developed primarily by the larger Japanese electronic musical instrument makers, it provided a language by which all devices constructed to certain standards could be interfaced. This allowed data to be transmitted into, out of, and/or through any compatible piece of equipment. MIDI works by integrating timing and system control commands with pitch and note parameter triggering commands, and sending them through a serial connection.[1] Although data is processed sequentially, the speed of transmission is usually fast enough to avoid noticeable auditory delays. The standard rate of transmission is 31.25 kbits per second, sent uni-directionally. This rather slow speed was agreed upon at a time when computer processor clock rates were only 1 or 2 MHz, but it has proved to be sufficient until this time. MIDI, while using much less processing power than digital audio, is only a set of instructions. Thus, its final output will sound different on every instrument which it plays, determined by the specific samples used by the software or hardware synthesizer in question.

Digital Audio is accomplished through a two part process, whereby analog sound is converted to digital data through sampling and quantization, and then, after recording, editing, and mixing, the process is reversed to return the data to the analog domain. The process has only a few steps, but the quality of the outcome it determined by many factors. The sample rate, the bit size, the consistency of the quantization size by the converters, and the speed and steadiness of the computer processor, are all factors which are vital in producing a high quality sound track. Unlike MIDI, which is a communication protocol, Digital Audio deals with actual sound samples.

This necessitates far more computer speed and disk capacity, but it results in a final product which has a fixed sound, although it is subject to the quality of the speakers or the headphones which project the signal.

Software Synthesis is the process by which oscillators and sound samples are programmed with sound altering information, through text statements or graphic controls, allowing the user to manipulate sound parameters in a manner similar to the physical changes which can be made with the knobs and sliders on a hardware synthesizer. The first successful instance of this process was accomplished by Max Mathews, in 1957, using the ground-breaking program, Music 1.[2] Today, the precision of software synthesis editing is virtually unlimited, because of the processing power of current computers, and the creative possibilities are equally beyond measure. I have used Csound to create many of the sounds used throughout the Suite. At times, software synthesis was used to make and to edit special effects, and at times it was used as the primary sound source for entire movements.

Introduction

The technical information presented in this work is closely tied to the musical content of *The Triangle Suite*. Throughout the process of writing the chapters which follow, I have made reference to my methods of organization, as well as to my technique in realizing a finished score. While at times the contents of the chapters on MIDI, digital audio, and software synthesis are of an explanatory nature, for the most part I have concentrated upon how these branches of music technology have been used to create the composition.

The three disciplines of technology discussed here were used in conjunction with each other for every movement except Movement V. That one movement was made within the software synthesis program Csound, and it is the only one which was not edited and mixed down in Pro Tools. However, it did have a MIDI origin, since the notes used in the eight voice orchestra were converted from a MIDI sequence before their parameters were edited in Csound.

Movements I, IV, and VI have an acoustic flute part which is performed with the stereo audio track. The flute part in Movement I is part of an orchestral ensemble, whereas the flute parts performed with Movements IV and VI are of a solo nature. In Movement IV, the flute part is totally improvisational, in the form of a blues solo. In Movement VI, the flute part is a dialog with an ostinato bass.

The orchestration for each movement is different, with the exception of Movements II and III, which both feature a MIDI woodwind quintet. Many of the same instruments appear in multiple movements, but the form of each movement is different. I tried to maintain an orchestral continuity, while varying the exact instrumentation.

In order to make *The Triangle Suite* universally performable, all six movements were formatted as stereo digital audio files. Without this step, the MIDI parts would have sounded different wherever they were played, because MIDI sounds are not really sounds at all, but commands given to a particular synthesizer. It is the samples stored within a particular instrument which determine the actual sounds. Regarding the Csound instruments, while it is possible to give real-time performances of the scores that it renders, the processing time to accomplish this feat makes it impractical in most instances.

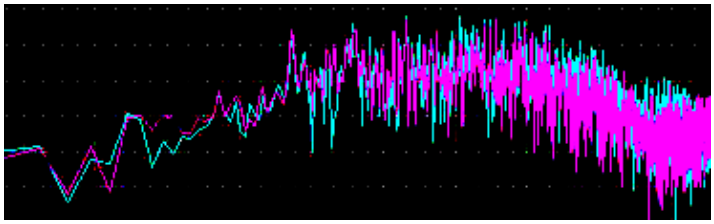
One of the nicest things about Csound, outside of its musical creativity, is that it is used by a fairly tight-knit community of musicians and computer programmers. These teachers, students, and composers from around the world freely distribute their knowledge, their instruments, and their compositions through websites and list servers. The following website and list server are two excellent places to start, since they provide a plethora of links to Csound programs, developers, users, and synthesized music.

www.csounds.com

program downloads for every platform, tutorials, books, instruments, music samples, utilities and related programs, and links to other sites

csound@lists.bath.ac.uk

list server for discussions and help using Csound. Caters to all levels of users, from beginners to programmers. Helpful, and good-spirited.



I. MIDI

MIDI Control

MIDI is a uni-directional transmission of instructional data at 31.25kbits per second. A separate link is need for bi-directional transmission. This speed, while rather slow by today's standards, was a convenient division of the 1 or 2 Mhz Master clock rate at its inception in 1982. Multiples of 31.25x2 = 62.5, 125, 250, 500, 1000, and 2000.[3] Messages are sent in bytes, with a start & a stop bit added to each transmission, thus an 8 bit message takes 10 bit periods to transmit.

Transmission takes place on a cable with five-pin DIN plugs. The inner-most three plugs are the functioning connections for most MIDI equipment. The hardware interface of most MIDI devices has three connectors; In, Out, and Thru. The In receives data from other devices; the Out transmits data to other devices; and the Thru passes incoming data directly out, without being processed or affected by the device. Thru boxes are used to speed up transmission when a system has many MIDI devices chained together. For optimal transmission, it is best to avoid cable lengths of more than 1 meter between devices.

While it not my intention to discuss MIDI with the depth reserved for software synthesis, I would like to briefly explain how MIDI instruments send and receive messages. Some MIDI devices have more features and abilities than others, but they share in common the same method of communication.

MIDI messages

There are five types of MIDI messages:

1.) Channel Voice Messages

Comprised of a status byte (beginning with a 1) and some number of data bytes (beginning with a 0). This leaves 7 bits, or a 0 – 127 range, for the least significant bits. The last 4 bits of the status byte are used to identify the channel, which can be 0 – 15. The first 4 bits identify the type of message.

Control Change Messages use a 2 byte identifier, and allow 128 separate control changes per channel.[4] *The upper 7 Control Messages are reserved for Channel Mode messages.*

2.) Channel Mode Messages
Beginning with the status byte B, these 7 places are reserved for controller reset, local on/off, all notes off, and the 4 modes of MIDI operation.
3.) System Common Messages
With status bytes ranging from F1 through F7, these messages do not include a channel number, because they are sent to all active devices.
4.) System Real-Time Messages
These messages are also sent without specific channel information. They are used to start and stop a sequencer, as well as to send a timing clock and a system reset message.
5.) System Exclusive Messages
These messages only send data to and from a specific device, defined by a manufacturer ID. All SysEx messages begin with F0, and end with F7. The message includes the type of data, the device ID, and any number of data bytes.

MIDI uses hex to transmit messages, which is convenient because it is a base 16 numbering system (0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F). This corresponds to the number of MIDI channels available, although devices may be chained together to multiply the amount of MIDI channels, creating 32, 48, 64, or more discrete channels. The specific message types, as well as their hex numbers for sending and receiving data, are usually provided in a table at the back of a MIDI device's manual.

Displaying and Editing MIDI information

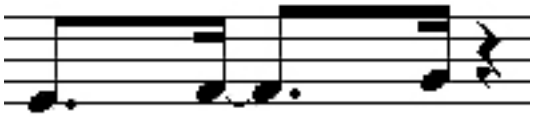
Software programs offer several distinct representations of musical data. It can be advantageous to have several of the following views open at once, in order to switch back and forth to the view which best suits the situation.

Staff view:

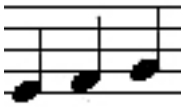
Individual notes, or blocks of events, can be entered, copied, pasted, and deleted while viewing any number of staves present in the score. Parameters, such as note length and volume, can also be edited here.

Quantization

Moving notes to a specific resolution, such as to the nearest 16th , 8th, or quarter note, can be very helpful. Programs also allow you to view a visually quantized score for editing purposes, while keeping the actual timing exactly as it was input. If the sequence is not visually quantized, a score representation is often difficult to read, since time accurate representations appear unlike the expected score. In addition to specific resolutions, programs allow you to choose degrees of "swing" 8th notes, as well as to randomly displace events by specified amounts for a less mechanical performance. Both notations in Example 1 sound exactly the same. They were entered in real time, played on a MIDI keyboard.



3 quarter notes, played unevenly, with varying lengths



The same 3 quarter notes, visually quantized, while sounding exactly the same

EXAMPLE 1

Event list view:

Individual notes and their parameters, program changes, and controller data are accessible here. Editing is usually done by typing in changes to the numbers associated with each event.

1	00:00:07:00	2:04:000	1	Control	66-Pedal (so	127	
1	00:00:08:00	2:04:118	1	Note	D 6	26	2:000
1	00:00:10:00	3:02:119	1	Note	Bb5	26	1:000
1	00:00:11:00	3:04:000	1	Note	A 5	27	119
1	00:00:12:00	3:04:119	1	Control	64-Pedal (su	0	
1	00:00:13:00	4:02:000	1	Note	Bb5	28	119
1	00:00:14:00	4:03:000	1	Note	D 6	29	1:119
1	00:00:16:00	4:04:119	1	Note	D#6	30	119
1	00:00:18:00	5:03:000	1	Control	7-Volume	64	
1	00:00:20:00	6:01:000	1	Note	A 6	32	1:119

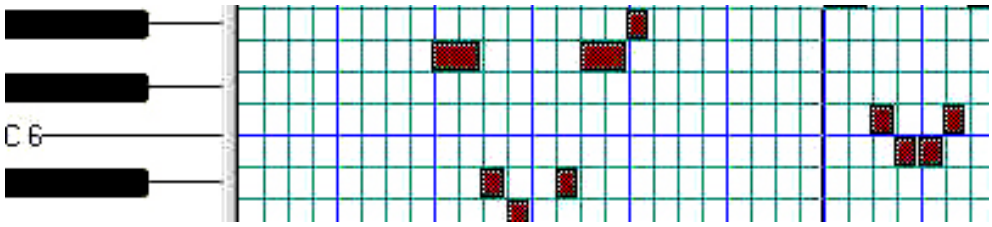
5

EXAMPLE 2

Grid or piano roll view:

Notes and controllers can be edited in real time during playback and recording. When looping part or all of a sequence, various

takes can be auditioned without stopping playback. Notes can also be moved, cut, copied, and pasted. Lengths of individual notes can be altered by dragging their beginnings or ends with a mouse.



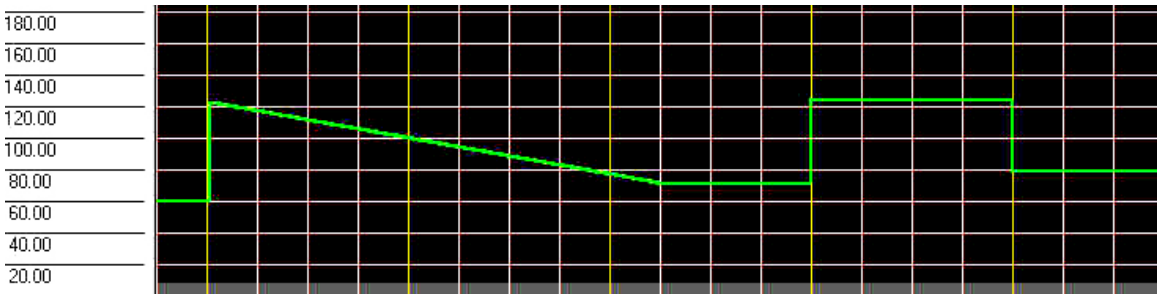
EXAMPLE 3 6

Track view:

The layout of a sequence is viewed here. Channel assignments, instrument names, port selection, and track transpositions are dealt with here. A score can also be sorted here, if relocating the tracks is desired.

Tempo view:

A graphic view of the tempo is represented by a horizontal line. Changes to the tempo can be made by clicking on a vertical position on the graph, either plus or minus the original tempo. Accelerandos and ritards can also be entered by drawing a line connecting two beat rates.



EXAMPLE 4 7

Mixer view:

A console view, with faders, panning, and effects processing is good for doing a final mixdown. Software mixer interfaces let

you mix MIDI data with a look similar to that of a digital audio workstation. Professional software products offer automated mixdown, which is recorded as an EDL (Edit Decision List) file. Another feature present on the mixing console is the ability to route sounds to different MIDI sources. This can increase the amount of channels, as well as the amount of sounds which are available. MIDI transmits on 16 channels, and each MIDI port is capable of discretely producing 16 tracks of MIDI sounds.

Other views which can be present, depending on the capabilities of the program, are *audio* and *video*. These views offer editing as well, but often not of the same depth as programs dedicated to audio and video processing.

MIDI velocity changes [8]

A. **Shifting** (scaling)
All velocities are shifted the same amount, either by a percentage of their value, or by a designation of a set amount, such as setting all notes to a specific number on the MIDI scale of 0 – 127.

B. Compression
The existing contour remains, but the volume of the notes selected are brought closer together or further apart from a center point. This allows you to maintain the shape of the dynamic curve, while confining or expanding the dynamic range.

C. Clamping
Working like a limiter or a noise gate, clamping acts only on notes which fall above or below a specified level. A positive value would limit notes from surpassing that level, and a negative value would insure that every note was brought up to that level.

D. Tapering
This has the effect of a crescendo or a decrescendo; also a fade in or a fade out. Two values are set, over the course to two or more note events. The dynamic values of those notes are changed by an equal amount. *Examples*: Three notes with values of 64, tapered from 64 to 70, would now have values of 64, 67, and 70, respectively. Seven notes with values of 64, tapered from 64 to 70, would have values of 64, 65, 66, 67, 68, 69, and 70, respectively.

Arranging Sequences

Sections of tracks can be named for easier identification. This is also more convenient to separate compositions into smaller sections for editing. This way, blocks can be copied and pasted to other locations, as in the case of a repeating chorus. It is also helpful to break compositions into smaller sections when changing parameters, such as shortening the note lengths of a section which should be played staccato. The name of the section appears in the track view, and can be renamed to indicate a new instance of the same data. (Chorus 1, Chorus 2, etc.) I employed this technique when editing the Minuet and The Scherzo of *The Triangle Suite*. The character of these movements changed between the opening sections and the Trio sections, in the articulation as well as in the thematic material.

Non-note Events

Control, Program Change, and System Exclusive

These are events, placed in event lists with notes and note parameters. They can likewise be edited by changing their times, or any other aspect, such as which controller is being used, or which program is being inserted. Scaling factors, in the case of controllers, can also be edited. Volume can be adjusted over a period of events to create a crescendo or a decrescendo, and pitch bend can be increased or lessened. Thinning of controller data is also useful, because effects like pitch bend and aftertouch tend to take a lot of memory if allowed to send messages at their highest resolution. System exclusive data can be loaded from or sent to a hardware synthesizer before a sequence begins, or while it is playing.

MIDI Orchestra Used In The Triangle Suite

All 6 movements of the Suite were imported from a notation program, Finale, into Cakewalk, a MIDI editing and sequencer program. The tempos, dynamics, and articulations were adjusted in this environment, before Movements I, II, III, IV, & VI were imported as discrete MIDI tracks into Pro Tools. (Movement V was converted to a text file, and opened in Csound) Here the sounds were recorded as digital audio tracks, and then panned and mixed down to stereo left and right tracks. Effects were also added to some tracks before mixdown. The details of this process are outlined and explained in Chapter IV., which deals with the production of the Suite.

By the nature of their individual parameters, each MIDI sound has a different ADSR envelope, and thus requires editing to produce uniformity with other instruments. Note events were lengthened to create a legato effect where desired, and notes durations were shortened to create a staccato effect. The trio of the Minuet had many notes shortened to create a better contrast with the lyrical sections which precede and follow it. In the Scherzo, the ascending chromatic scales in the flute part of were made more fluid by removing space between the notes, creating a portamento effect.

The MIDI instruments used throughout the composition are: *flute, oboe, Clarinet, Bassoon, Trumpet, French Horn, Trombone, violin, bass, tympani, vibraphone, and mallet.*

Although the orchestration varies from movement to movement, I used the same parameters, effects, and panning when each instrument was used, to simulate the characteristics and the placement of an actual orchestra.

II. DIGITAL AUDIO

Discrete Time Sampling

In order to create a digital representation of an analog audio signal, sound is converted by a three step process, called *ADC*, or analog to digital conversion. First, a low-pass filter removes any signal found to be more than one half of the sampling rate. This is necessary, because of the Nyquist frequency.* Next, the amplitude of the signal is measured at equally spaced intervals of time. These measurements are used to convert the continuous bandlimited signal of analog audio into a discrete sequence of digitally recognized samples. Finally, a quantizer gives a specific numeric value to each attained measurement.

Alaising

If the audio frequency being sampled is greater than half of the sampling frequency, false signal components are created. Anti-alaising and low-pass filters provide attenuation of these frequencies. Sampling at the highest rate possible will help to alleviate this problem, so the process of oversampling is often employed. This allows the anti-alaising filter to be set higher, because the Nyquist frequency is also higher. After the signal is in the digital domain, the normal sampling rate is returned to by filtering the signal to the Nyquist frequency, and at that point the extra samples created from the oversampling process are simply thrown out.

Quantization

Quantization is the value of, or the distance between, every sample measurement. An accurate representation of an audio signal is a direct result of the bit resolution used to obtain these measurements. The higher the bit rate, and the higher the sample rate, the less distance between each quantized interval. Even with CD or DVD standards, there will be some errors created by any sound that falls between these minute divisions. This will cause the generation of a small amount of noise.

***For complete signal reconstruction, the required frequency bandwidth is proportional to the signaling speed, and the minimum bandwidth must be equal to ½ the number of code elements per second.** [9]

Dither

Dither is the process of adding a small amount of noise to a quantized signal, performed prior to sampling. Its purpose is to eliminate distortion caused by quantization errors. It is especially helpful in masking degraded signals just above the level of silence. Errors in this area present the most degradation, because they are captured by the least significant bit. Dither smoothes over the 1-bit square waves that can result from this degradation.

DAC, or digital to analog conversion, reverses the *ADC* process. It is carried out by the procedure of turning on voltage generators to correspond to each bit of incoming digital data. First, the signal is converted to a time-varying voltage based to the sequence of quantized numbers created during the *ADC* process. Next, any glitches that were introduced by this process are eliminated by disregarding any fast transients. Finally, a low-pass filter set at half the sampling rate is used to smooth out any irregularities in the analog signal.

Removing unwanted noise

Two of the main advantages of digital music over analog music is that it can be edited quickly, and that it can be edited with more precision. Data can be cut and pasted to a new location without the necessity of physically cutting and pasting actual audio tape. The advent of EDL's has made it even easier; audio data doesn't even have to be actually changed at all. Directions are merely given to the computer to carry out, such as when and where to play, how loud to play, etc. Markers can be placed in precise locations, down to individual milliseconds. Elements can be cross-faded for seamless connections, and tracks can be bounced and mixed without building up noise and without loss of fidelity.

With all theses technical advances and advantages, music recorded in the digital domain still has problems to correct. One of the most important editing procedures is the elimination of unwanted noise. While there are many software and hardware processors dedicated to this purpose, it is first important to understand and classify the type of noise which may be present.

Impulsive Noise:

> This type of noise consists of transients, appearing like spikes on a waveform. An impulsive noise, resulting in a *click* or a *pop,* can be caused by a power surge, a bad audio connection, or an imprecise edit.

Continuous Noise:

> This is a constant sound, unrelated to the desired signal, such as hum from an air conditioner, or hiss built up from an analog signal. Continuous noise is often masked by the amplitude of the signal, becoming more obvious during quiet sections.

Harmonic Noise:

> Harmonic noise is also continuous, but it contains a fundamental frequency with harmonics. It can be caused by a ground loop, light dimmers, or AC power supplies.

Impulsive Noises need to be removed first, because they can contaminate the "noise print" that should be made during the continuous noise removal process. It's a good idea to try reversing the audio file to discover all impulsive noises. Often they have sharper back edges than front edges, making them easier to detect when an audio file is reversed. If software plugins are not able to remove impulsive noises, precise microedits to remove clicks and pops can be made by zooming in on and either deleting or redrawing the peak of the offensive sound.

Continuous Noises are difficult to remove, because source material can be damaged along with noise. A "noise print" is made by finding an area of the sound that contains only noise, usually at the beginning or end of the recording, or during a low amplitude section between passages. One second is enough. A waveform view will show the threshold of the continuous noise, which lets you to set the amplitude attenuation to just above the level of the noise print. If it's set too high, you may lose some wanted sound.

Harmonic Noises are also continuous noises, but they are related to the frequencies of an audio signal. They can be partially removed with the procedure for removing continuous sounds. For what remains, there are software plugins for hum removal or harmonic rejection. These processing tools resemble a string of notch filters strung together to cover multiple frequencies.

Sound effects, placement, and mixdown for The Triangle Suite

The sound effects used in The Triangle Suite are mostly outdoor sounds, depicting the atmosphere of 1911 New York City. There are the *timeless effects* of birds, crowds, and people walking; the *time specific effects* of carriages, trolleys, and horses on cobblestones; and finally the *event specific effects* of the fire and the escape from the building. The sounds are further distinguished by their use as either background ambience, or to denote specific events in the course of the afternoon of March 25th.

The majority of the sounds are from the sound effects library of the NYU music department, the remainder being from websites which specialize in the distribution of sound effects. Several sounds and instruments were also created in or processed with Csound.

After the orchestral sounds were mixed down to two tracks in Pro Tools, the effects were imported and placed on discrete tracks. Most of them needed to be shortened or lengthened to fit specific locations in the score. Some of them also needed to be resampled, in order to conform with the 44.1 kHz sampling rate used throughout the project. Fades were also added to smooth the start and the end times of the majority of the sounds.

I tried to find sounds effects which were quite specific in character, so I found it unnecessary to do a lot of sound editing. I did, however, work judiciously with panning and dynamic levels. Many of the sounds involved movement, which made shifts in placement an important element in the final production. They also needed to be placed where they would be understood in relation to the thematic material; in The Scherzo, for instance, footsteps climbing stairs are heard in conjunction with a flute part playing an ascending chromatic scale.

The specific settings for effects are detailed in Chapter IV, where an outline is presented of the approach I used in creating each movement of *The Triangle Suite.* At this point I would like to touch upon how the tracks were organized, and what digital processing was used to arrive at a final stereo mix for the finished movements.

Movement I.

The Overture captures the sounds of three compositions from 1911, blended together at times in a collage, but mostly as a medley. An excerpt from Scott Joplin's opera *Treemonisha* opens the Suite, following the sounds of city traffic, broken by horses walking on cobblestone. The actual scene of the tragedy was on Greene Street, which is still paved in

cobblestones. Panning was used to create the illusion of movement in the horses, as well as in the walking footsteps which are heard later.

An excerpt from Stravinsky's *Petrushka* is the next melody heard, used slowly to accompany the sounds of a factory. The sound effect is a background ambience to the plodding melody. The footsteps of women walking accompany this theme when it appears at a faster pace later in the movement.

The third theme is the chorus from a popular song, *"Every Little Movement Has A Meaning All Its Own"*, by Karl Hoschna. This was not only a hit song in 1911, but it was being sung by a few of the girls working in the factory at closing time, moments before the fire. A low murmur of people accompanies this theme, with a foreshadowing of what was soon to follow.

Movement II.

The sound of birds, happily chirping on an early Spring day opens the Minuet. This movement is classical in form, with repeated A and B sections, a Trio, and a Da Capo. I wanted this movement to show none of the turmoil and heartbreak which was to follow.

The birds were split on four tracks, and panned back and forth randomly before being mixed down to a stereo pair. Light laughter, and panned walking, are also present.

Movement III.

The Scherzo is the first indication of the fire at the Asch building, abruptly starting with the sound of a small explosion. The augmented chords, syncopations, and chromatic scales describe the sudden change of atmosphere. The crowd noise is more intense, the walking is faster, and the horses are galloping, which was included in accounts at the time, as Patrolman Meehan was seen arriving by horseback to the scene from nearby Washington Square Park. The accelerando that ends the movement is highlighted by the sounds of the valiant officer climbing the stairs of the building to aid in the rescue.

Movement IV.

The 5/4 blues progression that runs through this movement contains sounds of the fire, the noisy fire escape, and the crowd below. I refrained

from more graphic use of screams, or the sound of the 62 girls who jumped to their death that day. I tried to express the feeling of the event without being insensitive to the recent tragedy on September 11th.

The improvisational flute solo which pervades the movement is to be played in a frantic fashion, using descending scales and multiphonics to describe the scene. Although sound effects would have painted a clearer picture of the situation, I felt that tonal description would be more tasteful at this time.

Movement V.

This is the only movement created entirely in Csound, consisting of a twelve tone vocal melody poly-melodically placed over a brass quartet playing a slow hymn, *Praise God From Whom All Blessings Flow.* The distant sound of the hymn is often masked by the anguish of the constantly modulating voices, accelerating through most of the movement. There are sound parameters and effects used throughout the movement, which are described in Chapter IV. The entire Orchestra and Score files for this movement are found in Appendix 4.

Movement VI.

The epilog is a duet between an ostinato bass line and a solo flute, in a meter of 7/4. A few other MIDI instruments are in the orchestration, as well as three sounds created in Csound. These are *space, wind,* and a *vocal* track. The sound effects are brought in and out of the movement, panned in different ways for variety. The mood is one of reflection, as the name *Used To Be* suggests a look back at how things were before the tragedy.

Other than Movement V, which was rendered in Csound, the sound elements were divided into three sections; orchestra, sound effects, and ambience. These three aspects of the score were each mixed down to a stereo pair, and the final mixdown for performance was from 6 tracks to 2, accomplished in Pro Tools. The output audio files were performed live in *Sound Forge*, which was able to have all six movements open at once.

III. SOFTWARE SYNTHESIS

What Is Csound?

Synthesis is the creation of sound by electronic means, either from a microprocessor, or from some other type of electronic circuit. Software synthesis adds the further stipulation to the description that the process is accomplished with the aid of software. Csound is a software program which allows you the freedom to create sound by using text commands and arguments to specify the parameters of any one sound, or of any combination of sounds.

Csound is public domain software, modified by and contributed to by many users, written in C language. The instruments and the scores which it generates are compatible with all operating systems, although the specific interfaces used for various computer platforms vary slightly from each other. Upgrades to the program are provided several times each year, contributed to by a network of developers and users from around the world.

I have used Csound exclusively in Movement V of *The Triangle Suite*, as well as on numerous times in all the other movements to either create or enhance sounds. Sometimes I used it to create synthesized orchestral instruments, and at other times I used it to process sound samples I already had from other sources. I will discuss these uses in detail later in the chapter.

Csound: How It Works

Csound performances are generated from two text files, which reference each other: The *orchestra* file, with the extension *.orc*, and the *score* file, with the extension of *.sco*. Csound compiles and executes the commands and arguments which are entered into these files. The sound file it produces is a binary code representation of the sound.[10]

The orchestra file begins with a header, which describes several unchanging attributes of the instruments which follow. Although other statements may be added, the basic header must contain the these lines. Spaces between Csound statements are optional.

sr sampling rate
kr control rate
ksmps sr/kr
nchnls number of output channels (1, 2, 4)

EXAMPLE 5

The sampling rate **(sr)** is set at this point, most commonly 44.1 kHz. (Any sample rate which is compatible with a computer's sound system is permissible.) The control rate **(kr)** is usually substantially lower, so it can run faster when processing data that is not dependent upon a high sample rate for accurate sound representation, such as adding vibrato to a sound, or adjusting the parameters of a sound envelope. The third line notes the ratio between the sample rate and the control rate **(ksmps)**, and is found by dividing the sample rate by the control rate. The fourth line sets the number of channels, either 1, 2, or 4. The header channel setting must be adhered to in all instruments, which means that the output of each instrument must reflect the number of channels present in the header. If the orchestra header reads **nchnls = 2,** there must be exactly two outputs in every instrument.

Following the header is a listing of all the instruments which will be used to create the score. The first line is always **instr** followed by a number from 1 to 200. The last line is always **endin** which tells Csound that the instrument has ended its definition. Between these two lines are any number of parameters describing the sound, as well as sound samples which can be commanded to play either altered or unaltered from their present form.

Orchestra

instr (#)
any number of opcodes/variables/arguments/sound files
endin

EXAMPLE 6

The left-most entry is a variable, showing the result of an operation. If it begins with an "a", it is updated at the audio rate of the header. If it begins with a "k", it is updated at the control rate of the header, which is quicker because it reads less data per second. If it begins with an "i", it is only updated once, at the start of the instrument. If it begins with a "g" it is used globally, in more than one instrument, such as reverb or equalization. These variables will be shown in examples from *The Triangle Suite*.

The result of one operation can be used as an argument in a following operation. In Example 7, the opcode (operation code) *oscil* is used to create a sound with an amplitude of 10000, a pitch of 220 Hz, reading *f* (function) table 1. The audio signal (asig) is then sent out to either a speaker or to a disk, using the opcode *out*. The letters immediately following the variable letter are arbitrary. It is only the first letter which is crucial. It

is best to make the variable name define the function. In this case, “asig” is short for audio signal.

```
        instr 1 *

asig    oscil   10000, 220, 1
        out     asig
        endin
```

EXAMPLE 7

Score

f statements are used to create waveforms, or to list sample names

The following *f* table generates a sine wave, stating the number of the function, the time at which it will first be available, the size or depth of memory locations it will use, the GEN routine used, and the amount and strength of harmonic components present. In this case, only the fundamental frequency is represented.

```
        f1 0 4096 10 1

        ;Notes

        i1 0    3
        i1 4    3
        i1 6    1
```

EXAMPLE 8

Although other parameters (amplitude, pitch, attack, effects, pan, etc.) are usually present, the score shown in Example 8 says instrument 1 will play a note at time 0 for 3 seconds. Another instance of the same sound will occur at 4 seconds, following a 1 second rest. It will play for 3 seconds, and end at exactly the same time as the third note which begins at 6 seconds. The *f* table 1 will be set as an argument in the instrument line which contains the opcode used to produce the sound. (See Example 16, P. 28)

*** All Csound instruments and scores have been reproduced by exporting them in rich text format from *Csound Editor*, a front end interface for Windows, developed by Flavio Tordini, and available for free from his website.**

http://web.tiscali.it/_flat/index.jsp.html

GEN routines

f tables, or functions, are acted upon by GEN routines. Each of these routines is programmed with a different set of tables. For example, GEN 10 is set to specifically generate the harmonic partials of the frequency stated in an instrument's opcode argument, with the proportion of the numbers denoting the amplitude of each partial. In Example 9, *f* table 1 is initialized at time zero, with 4096 memory locations, telling GEN 10 to produce the first 5 partials of a frequency. The 2nd and the 4th partials would be twice as loud, and the 3rd partial would be 3 times as loud as partials 1 and 5.

```
f1 0 4096 10  1   2   3   2   1
```

EXAMPLE 9

The previous sine wave could be transformed into any other type of waveform, because this particular GEN routine is programmed to interpret the numbers to the right of the Gen number (10) as harmonic partials of a pitch designated in an orchestra file. By eliminating the even partials, the above table could be changed to produce a square wave.

```
f1 0 4096  10  1   0   3   0   1
```

EXAMPLE 10

Amplitude and frequency can be designated as separate *p* (parameter) fields in a score, and often appear as *p4* and *p5*, respectively. They may, however, appear in an orchestra or a score file as higher numbered parameters. The first three parameters: instrument, start-time, and duration are required, and their location in the score is unchangeable.

The exact amplitude of each note can be more easily handled by using the opcode *ampdb* to read raw amplitude values and convert them to decibel values.[11] This eliminates the need to deal with large numbers when expressing volume levels. If amplitude values higher than the values listed in Example 11 are found, an error warning *"Samples Out Of Range"* will be generated as the score is being rendered. The score will complete processing, but there will be clipping present. Csound not only tells you that the amplitude is too high, but it also tells you the number of samples which were out of range, the time it occurred, and in which instrument(s). Often it is the problem of combined instrument sounds, which by themselves are within the proper dynamic range.

ampdb conversion	raw amplitude value	ampdb conversion	raw amplitude value
90 (db)	31622.78	44	158.49
88	25118.86	42	125.89
86	19952.62	40	100.00
84	15848.93	38	79.43
82	12589.25	36	63.10
80	10000.00	34	50.12
78	7943.28	32	39.81
76	6309.57	30	31.62
74	5011.87	28	25.12
72	3981.07	26	19.95
70	3162.28	24	15.85
68	2511.89	22	12.59
66	1995.26	20	10.00
64	1584.89	18	7.94
62	1258.93	16	6.31
60	1000.00	14	5.01
58	794.33	12	3.98
56	630.96	10	3.16
54	501.19	8	2.51
52	398.11	6	2.00
50	316.23	4	1.58
48	251.19	2	1.26
46	199.53	0	1.000

12

EXAMPLE 11

When notating frequency, the exact pitch of equally tempered notes can be converted from a decimal representation (261.626 = middle C) to a more manageable 8.00 = middle C, through the use of the opcode *cpspch*, which instructs the notes of the octave to stand for the exact frequency desired. The table shown in Example 12 represents one octave of notes, represented as 12 equal divisions. Fractional changes to the *cpspch* (cycles per second to pitch) values can be entered to create intervals of any other size desired. To create a note ¼ tone above C4, the *cpspch* number would be noted as 8.005

note number	pitch	cpspch conversion
C4	261.626	8.00
C#4	277.183	8.01
D4	293.665	8.02
D#4	311.127	8.03
E4	329.628	8.04
F4	349.228	8.05
F#4	369.994	8.06
G4	391.955	8.07
G#4	415.305	8.08
A4	440.000	8.09
A#4	466.164	8.10
B4	493.883	8.11
C5	523.251	9.00

13

The 200 instruments which may be present in an orchestra include standard band and orchestral instruments, sound effects, and effects processing instruments. The efficiency of the statements which are created determines how quickly scores are rendered. There can be several paths which lead to the same output of sound, but there is usually a preferred way to enter the parameters and variables to provide a method which will use the least amount of processing power and time. Some experimentation is needed to expedite the rendering of scores. Csound can be slow when rendering complex scores, making real-time performance possible but difficult.

Single letter statements are used to specify global parameters, such as tempo, *t*, division of a score into sections, *s*, and a place to stop processing, *e*. They, and all other statements used in Csound, are listed in the easy to understand helpfile which accompanies the program.

t statements work in pairs, the first one designating a position in the score, the second one designating a tempo. The first one must be zero.

*t*0 72 3 80 5 120 5 60

EXAMPLE 13

The time statement shown in Example 13 will cause a score to begin at 72 bpm, and have a linear acceleration to 80 bpm after 3 seconds. At 5 seconds, an acceleration to 120 will have taken place. Also at 5 seconds, there will be an immediate change of tempo to 60 bpm. This tempo will

remain until the end of the score or section. Only one *t* statement is allowed in a section, but an unlimited number of tempo changes can be made.

s statements can be placed within a score to cause a new start immediately after that point.

```
i1 0    2
s
i2 0    2
```

EXAMPLE 14

Instrument 1 in Example 14 will play for two seconds and stop. Instrument two will then play for two seconds. Without the *s* statement, both instruments would start concurrently.

An *e* statement causes a stop to processing, which is useful when auditioning certain sections of a score. It eliminates the need to wait for the entire score to render. The score shown in Example 15 will stop after 2 seconds, without playing instrument 2.

```
i1 0    2
e
i2 3    5
```

EXAMPLE 15

Each opcode has its own syntax of arguments to be acted upon. There are usually a few arguments which must be present, along with others which are optional. In Example 16, *oscil,* output at a control rate variable, requires an amplitude, a frequency, and a function table (ifn = instrument function number). There is an option to indicate the phase of the oscillator. Optional arguments are always enclosed in brackets. The first line is a description of the necessary arguments; the second line is what the actual orchestra line might look like.

```
k1      oscil   kamp, kcps, ifn [,iphs]

k1      oscil   10000, 440, 1
```

EXAMPLE 16

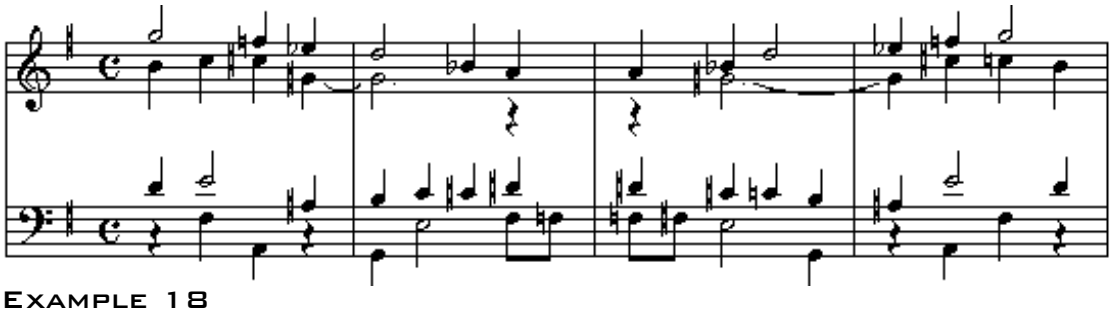
Global Instruments

Csound allows global instruments to be created, which can be called to act upon any or all instruments within an orchestra. Global instruments must begin with the letter *"g"*, and are used for reverb, equalization, or any other effect used with more than one instrument. Placing a *g* before any *i*, *k*, and *a* variables (*ga*reverb for instance) tells Csound that the parameters which follow refer to a global instrument at the specified variable rate. The rest of the name is arbitrary, in this case describing reverb. Example 17 shows a flute instrument processed with a global reverb. The same opcode, *reverb,* could have been used within the flute instrument, if reverb was not needed in any other instruments. In Csound, anything following a semi-colon is read as a comment, and is not acted upon.

```
        instr   1       ;Flute

idur            =       p3      ;duration
iamp            =       p4      ;amplitude
ifrq            =       p5      ;frequency
iskiptime       =       p6      ;amount of sample to skip
iattack         =       p7
idecay          =       p8
isustain        =       p9
irelease        =       p10
ibalance        =       p11     ;normalization
irvbgain        =       p12     ;reverb gain
kamp            linseg  0, .2, 1, .1, .8, p3-.5, .8, .2, 0  ;ADSR envelope
kdclk           linseg  0, .02, 1, p3-.1, 1, .02, 0         ;declick envelope
ar              linseg  10000, 142, 5000, 161, 0
k1              poscil  8, 4.2, 3                           ;4.2 Hz vibrato
asig            poscil  p4*.15, p5+k1, 3
arampsig        =       kamp+kdclk*asig
                outs    arampsig*ibalance, arampsig*(1-ibalance)
garvbsig        =       garvbsig+arampsig*irvbgain
        endin

        instr   2       ;GLOBAL REVERB

irvbtime        =       p4      ;reverbtime
asig            reverb  garvbsig, irvbtime
                outs    asig, asig
garvbsig        =       0       ;resets the reverb time to 0
        endin
```

EXAMPLE 17

Movement V of The Triangle Suite was the only one which was edited entirely in Csound. The *Funeral March* is made up of two separate quartets. The first one is a vocal quartet, singing a twelve tone melody which is followed by the same melody in retrograde. The melody then modulates through all twelve keys.



EXAMPLE 18

Underneath this melody, a brass quartet plays the hymn *"Praise God From Whom All Blessings Flow"* , augmented to be heard very slowly. Each of the Csound instruments was subdivided into pairs, allowing me to fine tune the parameters of the upper and the lower voices.

When I created each set of instruments, they sounded good by themselves, but when I put the vocal and brass parts together, the vocal parts got lost, even if I raised their dynamic levels. What I did, on the advise of Dr. Boulanger, was to give the attack of the vocal part more punch, so it would come through the heavier sound of the brass. Using the opcode *pluck*, the vocal attacks became more apparent. *Pluck* can be used to simulate pizzicato strings, but it also can be used to boost an attack, simulating the attack of a drum more than that of a string.

Most of the opcodes used to create instruments in Csound are multifunctional, depending on the parameters and the arguments they are fed. They all work in combination with each other, so often an instrument is made up of many opcodes. The opcodes are processed in the order in which they occur, so the output variable of one opcode is often an argument in a following opcode. Example 19 is the instrument I used to create the Soprano and the Alto voices. The audio output variables, beginning with the letter *a*, appear later in the instrument, as arguments fed to other opcodes, and finally to the output of the instrument. If a variable has been sent to another opcode, it doesn't have to be also sent to the output of the instrument.

```
        instr 1          ;SOPRANO & ALTO
iamp            =       p4*.6
i2              =       p6
i2              =       (i2/p5)+.5
i2              =       int(i2)
i4              =       p8
i5              =       p9*.4
i9              =       exp(1.5*log(p5/32767))
i13             =       i4*p5
i14              =      i9
irvbgain        =       p10
iEGgain         =       p11
k1              poscil  1.5, 2.7, 1       ;Vibrato: amp 1.5, 2.7 Hz, f1
a1              linen   20,.1,p3,.08
a2              oscil   i13,p5,3
a8              =       p5+a2
a6              linseg  -.03,.07,.03,.03,0,p3-.1,0
a6              =       a6+1.
a1              oscili  a1,(a8+a2)*a6,3
a7              =       a2+i2
a3              linseg  0,.07,.1,.03,1.,p3-.008,1,.02,.1,.03,0
a3              oscili  a3,a7*a6,3
ar              pluck   iamp*1.5, p5, p5, 3, 3, 0, 1
arampsig        =       ar*.3+k1*(a1+a3)*p5*.09
                outs    arampsig * ibalance, arampsig * (1 - ibalance)
garvbsig        =       garvbsig + arampsig * irvbgain
      endin
```

EXAMPLE 19

The lines *irvbgain* and *iEQgain* point to where the reverb gain and the equalization gain are found in the score. The next line uses *poscil* to create a vibrato, with the parameters commented on in green. The other opcodes shape the sound, and after using *pluck* to give the sound more presence, the signal is sent through the stereo *outs* and processed with reverb (*garvbsig)*.

Pluck requires several arguments: **ar pluck kamp, kcps, icps, ifn, imeth**

ar = *variable output at audio rate*
kamp = *control rate amplitude*
kcps = *control rate frequency*
icps = *audio rate frequency*
ifn = *function table used to process sound*
imeth = *method used. Pluck is used mostly to make drum or string sounds. There are six methods of processing sound.*

I experimented with different *f tables,* and with different methods of processing, before deciding upon the settings shown in Example 19. I always find it best to change one argument at a time, so I can hear exactly what is being changed. This is a good practice when building orchestras, because often it can become overwhelming to understand all that is going on when a score is rendered in Csound. If you change too many things at once, sometimes the score will not run at all, and it becomes difficult to pinpoint the problem. Csound does generate error messages while it processes, but the messages are not always specific enough to help.

In Movement VI, *"Used To Be"*, an acoustic flute solo is accompanied by several electronic instruments. They are mostly of MIDI origin, with the exception of a vocal sound, and two sound effects, which were all created in Csound. The sound effects employed the use of a phase vocoder and the process of convolution, which will be explained before describing the instruments.

Phase Vocoder

A phase vocoder performs time-scale and/or pitch transposition of sound samples. Used within Csound, control of individual harmonics is possible, through Short-Time Fourier Transform (STFT) analysis. The output of this method of analysis is virtually identical to the original input, since it is based on the harmonics of a specific signal. [14]

The signal produced by a phase vocoder is the sum of the sound waves found by a fixed bank of bandpass filters, the parameters of which are determined by the time-varying amplitude and frequency of each sine wave. The phase vocoder analysis of these parameters is expressed with a succession of overlapping Fourier transforms.[15] While it is possible to use a phase vocoder with any imaginable sound, the results are more predictable when processing sounds which have harmonically related frequencies, such as musical pitches.

Csound uses the opcode *pvoc* to manipulate the results of phase vocoder analysis, a process which must first be performed by *pvanal.* There are defaults for the possible settings to analyze a sample, but they can be changed. *Pvanal* converts a soundfile into a series of Short-Time Fourier Transform (STFT) frames at regular timepoints; a frequency-domain representation. The output file can be used by *pvoc* to generate audio fragments based on the original sample, with timescales and pitches arbitrarily and dynamically modified.[16]

The sample rate, the duration of the file, and where to begin the analysis, are all editable parameters. If no changes are entered, the program automatically performs at the sample rate of the file, analyzing the entire sample. The following parameters are critical to the output.

Framesize – This is the number of samples in each Fourier analysis frame, noted as a power of two, from 16 through 16,384. Using larger numbers will avoid "smearing" or reverberation, but if the framesize is too large, the time resolution may be inaccurate.
Windfact – This is the overlap factor, controlling the number of Fourier transforms per second. It expresses the length of the segment of the sample to which pvanal applies an envelop. By having a window overlap, it means that even though the framesize is set to a certain size, the analysis times are staggered to overlap each other. This helps to alleviate the conflict between frequency and time accuracy explained below.
Hopsize – This is the Fourier frame offset, specifying the increment between successive frames of analysis. Either windowsize or hopsize can be set; they can not both be specified on the same sample.

The framesize setting is inversely proportional to the precision of the frequency and the time resolution analysis. In other words, while a large framesize will increase the frequency accuracy, it will make the time resolution less precise. Experimentation is necessary to determine the best size for each analysis, based on the particular characteristics of the sample. The *windfact* setting allows large framesizes to more accurately interpret time resolutions by being analyzed at staggered intervals, rather than consecutively. Csound also tries to adjust for imperfect settings by interpolating the values it reads at every control rate period.[17] It is important to remember that each window returns only one frequency value per spectral component.

A good rule to follow in selecting the best framesize is to make it at least as large as the number of cycles required to cover one cycle of the lowest frequency found in the sample. [18] If the lowest frequency is 100 Hz, having a cycle of 441 times per second at a 44.1 kHz sample rate, a framesize of at least 512 would be required for an accurate result. 256 would be too small.

The hopsize specifies the increment by which the selection moves through the sound. Rather than having analysis windows lined up end to end, the time at which successive sample selections are analyzed can be defined to help alleviate the problems caused by the compromise between frequency

and time accuracy. A useful equation to help understand the definition of the hopsize is:

hopsize = *windowsize/overlap*
overlap = *windowsize/hopsize*

The opcode which performs phase vocoder re-synthesis is *pvoc*. The syntax used is: ar **pvoc** ktimpnt, kfmod, ifilcod [, ispecwp]

ar = *variable output at audio rate*
pvoc = *phase vocoder opcode*
ktimpnt = *when and which frame is read from the analysis file*
kfmod = *pitch transposition factor, processed at control rate*
ifilcod = *name of analysis file; in quotes, with .pvc extension*
ispecwp = *optional; preserves spectral envelope if 0*

In Example 20, the time value, *ktimpnt,* goes from 0 to 1 in the time stated in the score for p3 (duration). *Pvoc* takes that data and processes the analyzed file *"voice.pvc"*, sending out a signal 1 octave higher . The pitch change is instructed by the 2, which is the *kfmod* value. Arguments in instruments are separated by commas.

```
        instr 1
ktimpnt     line    0, p3, 1
ar          pvoc    ktimpnt, 2, "voice.pvc"
            out     ar
        endin
```

EXAMPLE 20

Convolution

Convolution is a point by point mathematical operation in which one function is effectively smeared by another. [19] Convolution is the combining of two signals; one audio signal with the impulse response of another. When the two waveforms are multiplied, their combined spectra become convolved.

The multiplication which takes place is in the frequency domain, which if different from ring modulation, which takes place in the time domain. These two processes, which are both used to filter two signals, are dissimilar in the way they arrive at their result, as well as in the domain in

which those results take place. Ring modulation multiplies the samples of file *A* by the samples of file *B*. Convolution multiplies every single sample in file *A* by every single sample in file *B*. This produces an array of values, the product of which will only contain those frequencies which are included in the spectra of both sounds.[20]

Csound uses the program *cvanal* to analyze a sound file, and to create its impulse response. As with *pvanal,* you may change the default settings. The main parameters are: sample rate, number of channels, at what point to begin analysis, and how long analysis is desired. Regarding the length of the convolution file, short impulse responses are generally more effective in producing musical results.

```
        instr 1
a1      diskin      "vocal.wav", 2
a2      convle      a1, "pianostr.con"
        outs        a2*.02, a2*.02
        endin
```

EXAMPLE 21

Example 21 shows how the Csound vocal instrument created for Movement VI of *The Triangle Suite* was convolved with the impulse response of a piano string, in order to create a dreamy atmosphere. The frequency bands found in both sounds were represented in the output, which was rendered by Csound, and written to a .wav file. The opcode *diskin* was used to read the audio file, which has the argument *2* after the sample name to make the sound play at twice its frequency, or an octave higher. C*onvle* is the opcode which does the convolution, smearing the audio file with the impulse response. The amplitude of audio signal *a2,* which is the result of *a1* multiplied by *"pianostr.con"*, is factored by .02 to keep the output from distorting. Clipping or distortion would occur because of the process of realizing multiplications of all common frequencies during the convolution process. The opcode *outs* requires two instances of the output argument, in agreement with the header's number of channels (*nchnls=2)*.

A similar technique was used to soften the sound of a siren in Movement IV, as well as to give it a more musical timbre. The siren was created with the first instrument shown in Example 22, which takes a pitch of 450 Hz, and makes a glissando to 800 Hz. The process is completed three times. The amplitude and declicking lines are added to provide a smoother sound. The waveform that Csound rendered from this instrument was used

with the same *pianostr.con* impulse response file pictured in Example 21. The result was the sound of a siren doing a musical gliss, rather than the harsh, non musical sound of a police or a fire siren.

```
        instr 1  ;gliss.orc
kglis           linseg  450, 2, 800, 2, 450, 2, 800, 2, 450, 2, 800, 2, 450
kamp            linseg  0, .2, 1, .1, .8, p3-.5, .8, .2, 0   ;ADSR envelope
kdclk           linseg  0, .02, 1, p3-.1, 1, .02, 0
a1              oscil   p4, kglis, 1
arampsig        =       kamp+kdclk*a1
                outs    arampsig, arampsig
        endin
```

Output of *gliss* convolved with *pianostr.con* as a separate operation.

```
        instr 1  ;glisspianocov.orc
a1              diskin  "gliss.wav", 2.2
a2              convle  a1, "pianostr.con"
                outs    a2*.007,a2*.007
        endin
```

EXAMPLE 22

Returning to the effects created for Movement VI, the first one is a *"space"* sound, which was used as a special effect during two of the verses. An audio sample was convolved with the impulse response of a Gaussian Reverb, creating a supernatural sound. The result of that process was then sent through a phase vocoder, lowering the pitch by about 2/3 of its frequency, changing *ifreqscale* from 1 to .35. Example 22 shows the process at the point where the phase vocoder was used (*pvoc*) to change the frequency (*ifreqscale)*. Samples and analyzed files always appear in quotes. Colors have no significance.

```
        instr   1       ;Space
ifreqscale      =       .35 ;(1 = exact pitch, .5 = octave lower, 2 = octave higher, etc.)
ispecwp         =       .5
ktime           line    0, p3, p3*.9
irvbgain        =       p4
ibalance        =       p5
apv             pvoc    ktime, ifreqscale, "universe.pvc", ispecwp
                outs    apv, apv
garvbsig        =       garvbsig*irvbgain
        endin
```

EXAMPLE 23

That result was processed with a delay line, before importing the effect into Pro Tools. The amount of delay is .8 seconds, after using *diskin* to read the sample. Example 24 is the instrument which processed the delay.

```
        instr 1
a1      diskin      "usedtobe-space.wav", 2
a2      delayr      .8
        delayw      a1
aout    =           a1+a2
        outs        a1*1.5, a2*1.5
        endin
```

EXAMPLE 24

The sound effect *"wind"* is based on instrument 28 from Chapter 1 of the Csound Book. [21] Several arguments were changed, and a glissando and a delay line were added. The attack and the release times of the random noise were changed, to create a more natural effect. *"Space",* described above, and *"wind"*, were used to create ambience in Movement VI, *"Used To Be."*

```
        instr   3       ;Wind
idur    =               p3
iamp    =               p4*.004
ifrq    =               p5
iatk    =               p6
irel    =               p7
icf1    =               p5
icf2    =               p5*2
ibw1    =               p8
ibw2    =               p9
kenv    expseg          .001,iatk,iamp,idur/6,iamp*.4,idur-(iatk+irel+idur/6),iamp*.6,
                        irel,.01
anoise  rand            ifrq
kcf     expon           icf1, idur, icf2
kbw     line            ibw1, idur, ibw2
afilt   reson           anoise, kcf, kbw, 2
kglis   linseg          50, 5, 1000, 5, 50, 5, 1000 ;glissando 50 hz to 1000 hz in 5 seconds
kdclk   linseg          0, .02, 1, p3-.1, 1, .02, 0
k1      oscili          5,5,1                   ;vibrato: width, speed, table
a1      delayr          2
        delayw          afilt
aout    =               a1+afilt+k1
        outs            kglis*kdclk+aout*kenv, kglis*kdclk+aout*kenv
                endin
```

EXAMPLE 25

IV. The Making of The Triangle Suite

The Suite has been a main focal point throughout this paper, but in this chapter I will outline how I went about realizing the six movements. It is presented as an outline, because that is the way it was approached and carried out. The outline was amended constantly, but it always served as a guide, allowing me to maintain a consistent plan for the composition.

The outline existed for the most part on my laptop computer, which was also used to perform the composition at NYU on May 19th, 2002. I kept careful notes of my progress, while I did the actual production work on the desktop computer attached to my studio.

The Triangle Suite for Flute and Electronic Instruments

I. *Overture*: time=5:17

Acoustic flute, 8 piece MIDI orchestra, sound effects, and software synthesis

Medley/Collage of 1911 compositions:

1.) *Treemonisha*, Scott Joplin;
measures 69-82 from Overture

2.) *Petrushka*, Stravinsky;
"A Group of Drunken Revelers Passes, Dancing", 30 measures

3.) *"Every Little Movement Has A Meaning All Its Own"* Karl Hoschna;
chorus, 16 measures

II. *Minuet*: time=4:02

MIDI woodwind quintet with sound effects and software synthesis

III. *Scherzo*: time=2:43

MIDI woodwind quintet with sound effects and software synthesis

IV. *T-D*: time=3:10

Blues in 5/4
Acoustic flute, 8 piece MIDI orchestra, sound effects, and software synthesis

V. *Funeral March*: time=2:50

Poly-melodic 8 part Csound orchestra
12 tone synthesized vocal quartet modulating through all 12 keys
Synthesized brass quartet playing the hymn, *"Praise God From Whom All Blessings Flow",* with augmentation of the parts

VI. Epilog; *Used To Be*: time=3:12

Acoustic flute, 8 piece MIDI orchestra, and software synthesis

Movements I., II., III., IV., VI

A. Notation entered into Finale
    1. Score formatted for instrumentation, key, and meter
    2. Parts entered
        a.) Auditioned with playback
        b.) Changes in orchestration made
    3. Dynamics, articulations, tempos added
B. Score saved as MIDI file
    1. MIDI 1 format
    2. Each part on a discrete track
C. MIDI files imported into Cakewalk at 480 ppq
    1. Tracks assigned to instrument patches
    2. Adjustments to tempos, dynamics, articulations
        a.) Overture
            1.) Ritards at end of Treemonisha sections
            2.) Crescendos and diminuendos added
            3.) Slurs added to 16^{th} note scales
            4.) Note lengths shortened on Petrushka sections
        b.) Minuet
            1.) Ritard at end of trio
            2.) Crescendos and diminuendos added
            3.) Note lengths shortened and lengthened
                a. Length of all notes shortened for separation
                b. Slurred notes edited individually and in groups

c.) Scherzo
1.) Ritards at section endings, accelerando at end
a. ritards inserted at specific measures
b. accelerando at end drawn with mouse
2.) Crescendos and diminuendos added
3.) Note lengths shortened and lengthened
a. Length of all notes shortened for separation
b. Slurred notes edited individually and in groups

d.) T-D
1.) Note lengths shortened and lengthened
2.) Dynamics added to each verse
3.) Swing groove applied to all parts

e.) Used To Be
1.) Ritard inserted in last measure
2.) Dynamics added
3.) Note lengths shortened and lengthened

3. All movements: Random time changes applied to start times of all notes to create a natural flow (+/- 5 ppq)

D. Finished MIDI files imported into Pro Tools
1. Markers placed at each section or verse of each movement
2. MIDI tracks recorded as discrete audio tracks for conformity of sound in all environments, on all computers
3. Pan applied uniformly in all movements, to simulate consistent placement of orchestral instruments
a.) Flute center
b.) Oboe & French Horn: left (35%)
c.) Clarinet/Bassoon: right (35%)
d.) Trumpet/Trombone/Flugal Horn: right (35%)
e.) Violin; left (35%)
f.) Bass: center
g.) Tympani: right (35%)
h.) Mallet: right (35%)
4. Effects applied uniformly in all movements, to simulate consistent orchestral instruments. Amounts of gain or attenuation are in parenthesis.

a.) Flute:

1.) Q 10 paragraphic equalizer

125 Hz (+4), 250 Hz (+4), 500 Hz (+4), 1000 Hz (+4), 2 kHz (-2), 4kHz (-2), 8 kHz (+2)

2.) D-Verb reverb

50% wet, 75 % diffusion, 2.0 sec decay, 19.9 kHz high cut, 20 kHz LP filter

b.) Oboe:

1.) Q 10 paragraphic equalizer

125 Hz (+3), 250 Hz (+4), 500 Hz (+5), 1000 Hz (+4), 2 kHz (+3), 4kHz (+2)

2.) D-Verb reverb

50% wet, 75 % diffusion, 1.5 sec decay, 19 kHz high cut, 20 kHz LP filter

c.) Clarinet:

1.) Q 10 paragraphic equalizer

62 Hz (+1.5), 125 Hz (+3.5), 250 Hz (+1), 500 Hz (+3.5), 1000 Hz (+4.5), 2 kHz (-2), 4kHz (+2.5)

2.) D-Verb reverb

50% wet, 75 % diffusion, 1.5 sec decay, 12.7 kHz high cut, 20 kHz LP filter

d.) Bassoon:

1.) Q 10 paragraphic equalizer

62 Hz (+1.5), 125 Hz (-1), 250 Hz (+2), 500 Hz (+1), 1000 Hz (+3), 2 kHz (-1), 4kHz (+2)

e.) Trumpet:

1.) Q 10 paragraphic equalizer

62 Hz (+3), 125 Hz (+2), 250 Hz (+1), 500 Hz (+2.5), 1000 Hz (+2), 2 kHz (+1), 4kHz (+1)

2.) D-Verb reverb

50% wet, 75 % diffusion, 1.0 sec decay, 15 kHz high cut, 20 kHz LP filter

f.) Flugal Horn:

1.) Q 10 paragraphic equalizer

125 Hz (+1), 250 Hz (+1.5), 500 Hz (+5.3), 1 kHz (+5), 2 kHz (+1), 4 kHz (+2)

g.) French Horn:

1.) Q 10 paragraphic equalizer

62 Hz (+3), 125 Hz (+2), 250 Hz (+1), 500 Hz (+2.5), 1000 Hz (+2), 2 kHz (+1), 4kHz (+1)

h.) Trombone:

1.) *Q 10 paragraphic equalizer*

125 Hz (+4), 250 Hz (+3), 500 Hz (+2.5), 1000 kHz (+1), 4kHz (+3)

i.) Vibraphone:

1.) Q 10 paragraphic equalizer

62 Hz (-1), 125 Hz (+1), 250 Hz (+2.5), 1000 Hz (+2)

j.) Violin:

1.) Q 10 paragraphic equalizer

125 Hz (+2), 250 Hz (+3), 500 Hz (+3), 1000 Hz (+2), 2 kHz (+1), 4kHz (+1)

k.) Bass:

1.) Q 10 paragraphic equalizer

34 Hz (+18), 39 Hz (-15.3), 71 Hz (+2), 285 Hz (1.8), 475 Hz (-.5)

l.) Tympani:

1.) Q 10 paragraphic equalizer

125 Hz (+4), 250 Hz (+3), 500 Hz (+2.5), 1000 kHz (+1), 4kHz (+3)

m.) Mallet:

2.) Q *10 paragraphic equalizer*

125 Hz (+4), 250 Hz (+3)

E. Tracks mixed to stereo left and right

F. Sound effects tracks added to score
   1. Placed on individual tracks
   2. Each sound mixed and panned to mono or stereo
   3. Effects added

G. Background ambience: Mvt. I, II, III, IV
   1. Placed on individual tracks
   2. Each sound mixed and panned to 1 or 2 tracks
   3. Effects added

H. Background ambience: Mvt. VI, *"Used To Be"* (Created in Csound)
   1. *"Space"* sound was created using convolution. A sample rendered by processing two vocal speech samples from the Csound catalog of instruments (Costello *space*) was convolved with the impulse response from a Guassian reverb. The result of that was sent through a phase vocoder, and then processed with .8 seconds of delay. The times and the frequencies were adjusted to create an interesting ambience. The result was imported into Pro Tools, where panning and dynamic adjustments were made.
   2. *"Wind"* sound was created from an adaptation of part of instrument 28 from chapter 1 of the Csound book. Several arguments were removed before adding a glissando and a delay. A declicking line was also added to eliminate transient sounds, and the attack and release times were lengthened for a more natural occurrence of wind. The result was imported into Pro Tools, where panning and dynamic adjustments were made.

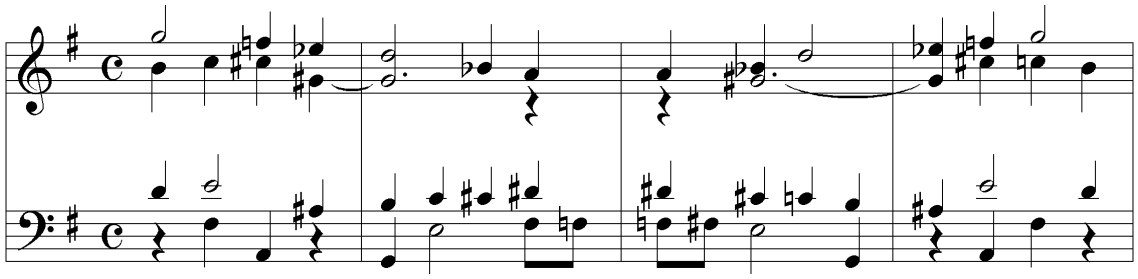
3. *"Vocal"* sound was based upon the soprano and alto voices used in Movement V, the *"Funeral March"*. That instrument was scored with a sustained, two part harmony, used as background to certain verses of the flute solo. The rendered wave file was convolved with the impulse response of a piano string. The result of the convolution was imported into Pro Tools, where panning and dynamic adjustments were made.

I. Final mix to stereo (6 to 2) : Orchestra, Effects, Ambience
   1. Plugin effects
   2. Automated mixdown for levels

Movement V.

The Csound orchestra and score files for the "Funeral March" can be found in appendix 4

The following example is the 12 tone melody assigned to the vocal instruments created with Csound.

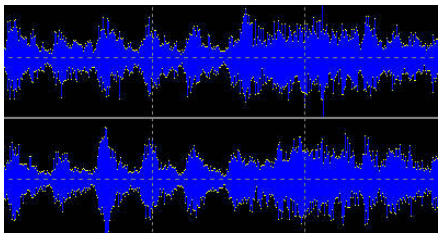


EXAMPLE 23

The 12 tone melody reads vertically and horizontally. Horizontally, measures one and two have independent 12 tone melodies; one in each staff. These melodies are then played in retrograde, after which time the four measures modulate through all 12 keys. Vertically, each measure is comprised of an independent 12 tone row. The soprano line is a natural minor scale with no fourth. The alto and tenor lines are mostly chromatic, and the bass line varies between stepwise motion and large leaps.

A. Notation entered into Finale
   1. 12 tone vocal S.A.T.B.
   2. Brass Quartet

B. Scored saved as MIDI file on 8 tracks
    1. MIDI 1 format
    2. S.A.T.B. (4 tracks)
    3. Trumpet, French Horn, Trombone, Tuba (4 tracks)
C. MIDI file imported into Cakewalk at 480 ppq
    1. Tempo set
    2. Adjustments made to dynamics, articulations
        a.) Dynamics for vocal: start < middle > end
        b.) Dynamics for brass: start > middle < end
        c.) Articulation for vocal: legato
        d.) Articulation for brass: lengths shortened for separation
D. MIDI to Csound (software: MidiFile2CsoundScore by Young Choi)
    1. Each of the 8 tracks converted to Csound format with 7 parameters: Instrument, start times, duration, amplitude, frequency, reverb time, equalization
    2. All tracks combined in 1 score file
        a.) Instrument 1; soprano and alto
        b.) Instrument 2; tenor and bass
        c.) Instrument 3; trumpet, French horn
        d.) Instrument 4; trombone, tuba
    3. Score opened in Csound Editor (Flavio Tordini)
    4. Global instruments added: Reverb, Equalization
E. Instruments created for score
    1. Soprano, Alto, Tenor, Bass: Parameters/Effects
        a.) Vibrato
        b.) Equalization
        c.) Reverb
    2. Trumpet, French Horn, Trombone, Tuba: Parameters/Effects
        a.) Vibrato
        b.) Equalization
        c.) Reverb



Conclusion

Completing this thesis, in conjunction with composing *The Triangle Suite*, has been a great experience. The work was hard, but it was the perfect way for me to conclude my studies at NYU. It made me review all that I had learned, as well as to incorporate new knowledge in my understanding of music technology. The field is so complex, and ever-growing, that this gave me the opportunity to solidify the concepts, definitions, and practical applications I had been introduced to during three enjoyable years of study.

I wanted to do something that had meaning to me, which I why I chose to use music technology to express myself about an important event in history. The scope and the dimensions of the project were I hope tastefully diminished because of the obvious parallels between The Triangle Shirtwaist Fire, and the attack of September 11th. My original plans to include video of the actual event were curtailed, as was my use of sound effects which explicitly depicted the events of March 25th, 1911.

These apparent shortcomings were at first a disappointment to me, but as I developed the score, using sound effects and Csound synthesis, I realized that I was not really sacrificing authenticity at all, but merely letting my audience members use more of there imaginations to understand the programmatic nature of the composition.

A good example of this was the Csound *siren* I created for Movement IV. Instead of using a blaring siren, which would have been an anachronism as well as an overly dramatic effect, I made my own siren, by programming an oscillator to rise and fall in pitch, employing the opcode *linseg.* (P. 35-36) I then convolved that sound with the impulse response of a piano string, to wind up with a piano glissando smeared with a rising and falling oscillator. The result was an interesting musical effect, which expressed the meaning I wanted to portray.

It was moments like that which taught me the most about how to use technology in an artistic way. Although my main objectives for this project were founded in technology, my goal was to discuss technology as it pertained to its functional use within a creative composition.

Following the premiere performance of *The Triangle Suite* at NYU's Frederick Loewe Theatre on May 19th, the contents of this paper will be posted in .pdf format on my website www.jlpublishing.com Streaming audio of the performance will also be available at that time. I welcome comments and questions regarding the paper or the composition. lee@jlpublishing.com

Appendix 1: Glossary

ADC – analog to digital conversion
ADSR – attack, decay, sustain, release
Alaising – distortion from recording of frequencies greater than ½ of the sampling rate
Analog – reproducing sound with voltage fluctuations analogous to the sound's pressure fluctuations
Augmentation – proportioned increase of note lengths
Bpm – beats per minute
Clipping – distortion from amplitude exceeding available level
Controller – physical controls or commands to change MIDI sound parameters
Csound – software synthesis program
- *Convolution* – frequency domain smearing of two sounds
- *Cvanal* – see P. 35
- *Ftable* – function table for reading samples or specifying harmonics
- *Gen* – routine specifying behavior of function tables
- *Header* – audio and control rates, and number of channels
- *Impulse response* – amplitude map of FFT analysis for convolved sound
- *Opcode* – operation code
- *Orc* – orchestra file
- *Parameter* – any sound variable
- *Phase Vocoder* – see P. 32
- *Pvanal* – see P. 32
- *Sco* – score file
- *Software synthesis* – creation of sound by computer program
- *Variable* – output of operation, at audio or control rate

DAC – digital to analog conversion
Delay – repeated audio signal, briefly after initial occurrence
Digital – binary representation of sound, discretely, through sampling
DIN – Deutche Industrie-Norm, standard of MIDI transmission
Dither – noise added to quantized sound for masking low level distortion
DSP – digital signal processor
EDL – edit decision list
Envelope – parameters of sound shape, most commonly ADSR
Equalization – attenuation or boost of individual frequencies by filters
Filter – function of removing or altering specific frequencies to change sound qualities
FFT – fast Fourier transform
Hex – (hexadecimal) 0-9, & A-F system of numbering. See MIDI P. 10
Interpolation – reconstructive filtering by retaining previous value
MIDI – Musical Instrument Digital Interface
Nyquist Frequency – see bottom P. 16
Oscillator – hardware or software sound producing object
PPQ – pulses per quarter note
Quantize (MIDI) – see P. 11
Quantize (sampling) – see P. 16
Reverb – simulated sound reflections
Sampling – digitally recording sound
Sequence – time ordered events
STFT – short-term Fourier transform
System Exclusive – instrument specific MIDI message sent by using manufacturer's ID number
Transient – sudden peak in amplitude
Waveform – digital audio in a format which can be read by and edited with computer software

Appendix 2: Examples

Example 1	Quantized notation	P. 11
Example 2	MIDI event list	P. 11
Example 3	MIDI piano roll view	P. 12
Example 4	MIDI tempo view	P. 12
Example 5	Csound header	P. 22
Example 6	Csound instrument format	P. 23
Example 7	Csound instrument	P. 24
Example 8	Csound score	P. 24
Example 9	Csound function table, sine wave	P. 25
Example 10	Csound function table, square wave	P. 25
Example 11	Csound amplitude conversion chart	P. 26
Example 12	Csound frequency conversion chart	P. 27
Example 13	Csound time statement	P. 27
Example 14	Csound section statement	P. 28
Example 15	Csound end statement	P. 28
Example 16	Csound opcode arguments	P. 28
Example 17	Csound flute instrument	P. 29
Example 18	Movement V theme, notation	P. 30
Example 19	Csound vocal instrument	P. 31
Example 20	Csound phase vocoder instrument	P. 34
Example 21	Csound vocal/pianostring convolution	P. 35
Example 22	Csound siren/pianostring convolution	P. 36
Example 23	Csound space instrument, Movement VI	P. 36
Example 24	Csound delay opcode	P. 37
Example 25	Csound wind instrument, Movement VI	P. 37

Appendix 3: Sound Effects

Movement I. *Overture*	crowd factory ambience horses on cobblestone park ambience traffic walking trolley
Movement II. *Minuet*	birds crowd walking laughing ambience horses on cobblestone trolley
Movement III. *Scherzo*	explosion horses running running up stairs walking crowd
Movement IV. *T-D*	clanging crowd fire engine siren horses running fire fire escape
Movement V. *Funeral March*	no effects
Movement VI. *Used To Be*	trolley birds Csound effects – space, wind, vocal

Appendix 4: Csound Orchestras & Scores

Movement V, *Funeral March*

Orchestra

```
;Lee Zakian, 2002
;Movement 5, Triangle Suite For Flute & Electronic Instruments
;FUNERAL MARCH

sr              =       44100
kr              =       4410
ksmps           =       10
nchnls          =       2
garvbsig        init    0
gaEQ            init    0
gaEQ2           init    0


        instr 1         ;High Vocals
iamp            =       p4*.5
i2              =       p6
i2              =       (i2/p5)+.5
i2              =       int(i2)
i4              =       p8
i5              =       p9*.4
i9              =       exp(1.5*log(p5/32767))
i13             =       i4*p5
i14             =       i9
ibalance        =       p7
irvbgain        =       p10
iEQgain         =       p11
k1              poscil  1.5, 2.7, 1             ;Vibrato: amp 1.5, 2.7 Hz, f1
a1              linen   20,.1,p3,.08
a2              oscil   i13,p5,3
a8              =       p5+a2
a6              linseg  -.03,.07,.03,.03,0,p3-.1,0
a6              =       a6+1.
a1              oscili  a1,(a8+a2)*a6,3
a7              =       a2+i2
a3              linseg  0,.07,.1,.03,1.,p3-.008,1,.02,.1,.03,0
a3              oscili  a3,a7*a6,3
ar              pluck   iamp*1.5, p5, p5, 3, 3, 0, 1
arampsig        =       ar*.3+k1*(a1+a3)*p5*.08
                outs    arampsig * ibalance, arampsig * (1 - ibalance)
gaEQ            =       gaEQ+arampsig*iEQgain
garvbsig        =       garvbsig + arampsig * irvbgain
        endin
```

```
        instr 2         ;Low Vocals
iamp            =       p4*.034
i2              =       p6
i2              =       (i2/p5)+.5
i2              =       int(i2)
i4              =       p8
i5              =       p9*.4
i9              =       exp(1.5*log(p5/32767))
i13             =       i4*p5*2
i14             =       i9
ibalance        =       p7
irvbgain        =       p10
iEQgain         =       p11
k1              poscil  1.5, 2.1, 1             ;Vibrato: amp 1.5, 2.1 Hz, f1
a1              linen   20,.1,p3,.08
a2              oscil   i13,p5,3
a8              =       p5+a2
a6              linseg  -.03,.07,.03,.03,0,p3-.1,0
a6              =       a6+1.
a1              oscili  a1,(a8+a2)*a6,3
a7              =       a2+i2
a3              linseg  0,.07,.1,.03,1.,p3-.008,1,.02,.1,.03,0
ar              pluck   iamp*1.5, p5, p5, 3, 3, 0, 1
arampsig        =       ar*.3+k1*(a1+a3)*p5*.035
                outs    arampsig * ibalance, arampsig * (1 - ibalance)
gaEQ            =       gaEQ+arampsig*iEQgain*.05
garvbsig        =       garvbsig + arampsig * irvbgain
        endin
        instr 3         ;Upper Brass
idur            =       p3
iamp            =       p4*.007
ifenv           =       5       ;brass settings:
ifdyn           =       5       ;amp and index envelope see flow chart
ifq1            =       p5
if1             =       1       ;duration ca. .6 sec
ifq2            =       p5*.095
if2             =       1
imax            =       5
ibalance        =       p6
irvbgain        =       p7
iEQgain         =       p8
aenv            oscili  iamp, 1/idur, ifenv     ;envelope
adyn            oscili  ifq2*imax, 1/idur, ifdyn ;dynamic
amod            oscili  adyn, ifq2, if2         ;modulator
a1              oscili  aenv, ifq1+amod, if1    ;carrier
k1              poscil  2, 2, 1                 ;Vibrato: amp 2, 2 Hz , f1
arampsig        =       a1*k1*p5*.009
                outs    arampsig * ibalance, arampsig * (1 - ibalance)
gaEQ2           =       gaEQ2+arampsig*iEQgain*.04
garvbsig        =       garvbsig + arampsig * irvbgain
        endin
```

```
        instr 4         ;Lower Brass
idur            =       p3
iamp            =       p4*.03
ifenv           =       5       ;brass settings:
ifdyn           =       5       ;amp and index envelope see flow chart
ifq1            =       p5
if1             =       1       ;duration ca. .6 sec
ifq2            =       p5*1.005
if2             =       1
imax            =       5
ibalance        =       p6
irvbgain        =       p7
iEQgain         =       p8
aenv            oscili  iamp, 1/idur, ifenv         ;envelope
adyn            oscili  ifq2*imax, 1/idur, ifdyn    ;dynamic
amod            oscili  adyn, ifq2, if2             ;modulator
a1              oscili  aenv, ifq1+amod, if1        ;carrier
k1              poscil  2, 2, 1                     ;Vibrato: amp 2, Hz 2 , table 1
arampsig        =       a1*k1*p5*.02
                outs    arampsig * ibalance, arampsig * (1 - ibalance)
gaEQ2           =       gaEQ2+arampsig*iEQgain*.05

garvbsig        =       garvbsig + arampsig * irvbgain
        endin
```

```
        instr 98        ;GLOBAL PARAMETRIC EQ for Vocals
iEQlevel        =       p8
ifc             =       p4              ;Center/Shelf (EQ band)
iq              =       p5              ;Q factor: (.2 to .003)
                                        ;High # =  narrow band width
                                        ;Low #  =  wide band width
iv              =       ampdb(p6)       ;Volume Boost/Cut: (12 to -12)
imode           =       p7              ;0=peaking, 1-low shelf, 2=high shelf
kfc             linseg                  ifc*2, p3, ifc/2   ;declicking
a1              =                       (p4*p8)*(p5+p7)*.001
aout            pareq                   a1, kfc, iv, iq, imode      ;parametric EQ
abalance        balance                 aout*gaEQ*.02, aout*gaEQ*.02
                outs                    abalance, abalance
gaEQ    =       0
        endin
```

```
        instr 99        ;GLOBAL PARAMETRIC EQ for Brass
iEQlevel        =       p8
ifc             =       p4              ;Center/Shelf (EQ band)
iq              =       p5              ;Q factor: (.2 to .003)
                                        ;High # =  narrow band width
                                        ;Low #  =  wide band width
iv              =       ampdb(p6)       ;Volume Boost/Cut: (12 to -12)
imode           =       p7              ;0=peaking, 1-low shelf, 2=high shelf
kfc             linseg                  ifc*2, p3, ifc/2   ;declicking
a1              =                       (p4*p8)*(p5+p7)*.001
aout            pareq                   a1, kfc, iv, iq, imode      ;parametric EQ
abalance        balance                 aout*gaEQ2*.02, aout*gaEQ2*.02
                outs                    abalance, abalance
gaEQ2 =         0
        endin

        instr 100
irvbtime        =       p4
asig            reverb  garvbsig,  irvbtime     ;put global sig into reverb
                outs    asig, asig
garvbsig        =       0                       ;then clear it
        endin
```

Score

```
;Lee Zakian, 2002
;Movement 5, Triangle Suite For Flute & Electronic Instruments FUNERAL MARCH

f1 0 8192 10 1
f2 0 8192 9 .25 10 .5 10  .25     ;TENOR, BASS (based on Dodge 411b)
f3 0 8192 9 1 1 0                 ;SOPRANO, ALTO (based on Dodge 411b)
;carriers
f4  0  512  10  1  ;BRASS (based on ACCCI: 20_10_3 from Amsterdam Collection)
;envelopes
f5 0 513 7 0 80 1  85 1  290 .8 63 0     ; amplitude & index envelope
;tempo: accelerate from 60bpm to 120bpm, drop to 60bpm, ritard to 40bpm
t       0       60      127     120     127     60      173     60      197     40

;10 BAND EQUALIZER for Vocals   .2 to .003       12 to -12       0,1,2
;Inst   Start   Dur     FrqCenter       Quality         Boost/Cut       Mode    EQgain
i98     0       197     100             .02             1               0       7
i98     0       .       200             .02             2               .       .
i98     0       .       300             .02             3               .       .
i98     0       .       400             .02             3               .       .
i98     0       .       600             .02             4               .       .
i98     0       .       800             .02             4               .       .
i98     0       .       1200            .02             3               .       .
i98     0       .       1600            .02             2               .       .
i98     0       .       2000            .02             1               .       .
i98     0       .       4000            .02             1               .       .
```

;**10 BAND EQUALIZER for Brass** .2 to .003 12 to -12 0,1,2

;Inst	Start	Dur	FrqCenter	Quality	Boost/Cut	Mode	EQgain
i99	0	197	100	.02	2	1	4
i99	0	.	200	.02	3	.	.
i99	0	.	300	.02	4	.	.
i99	0	.	400	.02	4	.	.
i99	0	.	600	.02	4	.	.
i99	0	.	800	.02	3	.	.
i99	0	.	1200	.02	2	.	.
i99	0	.	1600	.02	1	.	.
i99	0	.	2000	.02	-1	.	.
i99	0	.	4000	.02	-1	.	.

;**GLOBAL REVERB**

i100	0	197	3.0

;**SOPRANO**

;Inst	Start	Dur	Amp	Freq	P6	Bal	P8	P9	Rvb	EQ
i1	3.836	1.918	2500	783.991	700	.5	.25	5	.5	5
i1	5.764	0.95	2600	698.456	700	.5	.25	5	.5	5
i1	6.726	0.948	2600	622.254	700	.5	.25	5	.5	5
i1	7.674	1.92	2600	587.33	700	.5	.25	5	.5	5
i1	9.596	0.958	2600	466.164	700	.5	.25	5	.5	5
i1	10.566	0.948	2700	440	700	.5	.25	5	.5	5
i1	11.518	0.956	2800	440	700	.5	.25	5	.5	5
i1	12.486	0.948	2800	466.164	700	.5	.25	5	.5	5
i1	13.446	1.908	2900	587.33	700	.5	.25	5	.5	5
i1	15.36	0.954	3000	622.254	700	.5	.25	5	.5	5
i1	16.314	0.96	3100	698.456	700	.5	.25	5	.5	5
i1	17.284	1.91	3100	783.991	700	.5	.25	5	.5	5
i1	19.204	1.91	3200	880	700	.5	.25	5	.5	5
i1	21.124	0.95	3400	783.991	700	.5	.25	5	.5	5
i1	22.078	0.956	3400	698.456	700	.5	.25	5	.5	5
i1	23.042	1.912	3500	659.255	700	.5	.25	5	.5	5
i1	24.956	0.958	3600	523.251	700	.5	.25	5	.5	5
i1	25.92	0.954	3600	493.883	700	.5	.25	5	.5	5
i1	26.876	0.958	3700	493.883	700	.5	.25	5	.5	5
i1	27.842	0.952	3700	523.251	700	.5	.25	5	.5	5
i1	28.794	1.92	3800	659.255	700	.5	.25	5	.5	5
i1	30.726	0.948	3900	698.456	700	.5	.25	5	.5	5
i1	31.676	0.958	3900	783.991	700	.5	.25	5	.5	5
i1	32.634	1.92	3900	880	700	.5	.25	5	.5	5
i1	34.558	1.916	4100	830.609	700	.5	.25	5	.5	5
i1	36.486	0.948	4200	739.989	700	.5	.25	5	.5	5
i1	37.444	0.95	4200	659.255	700	.5	.25	5	.5	5
i1	38.394	1.92	4300	622.254	700	.5	.25	5	.5	5
i1	40.314	0.96	4400	493.883	700	.5	.25	5	.5	5
i1	41.276	0.958	4500	466.164	700	.5	.25	5	.5	5
i1	42.24	0.954	4500	466.164	700	.5	.25	5	.5	5
i1	43.202	0.952	4600	493.883	700	.5	.25	5	.5	5
i1	44.166	1.908	4700	622.254	700	.5	.25	5	.5	5

i1	46.082	0.952	4700	659.255	700	.5	.25	5	.5	5
i1	47.034	0.96	4800	739.989	700	.5	.25	5	.5	5
i1	47.996	1.918	4800	830.609	700	.5	.25	5	.5	5
i1	49.92	1.914	5000	932.328	700	.5	.25	5	.5	5
i1	51.842	0.952	5100	830.609	700	.5	.25	5	.5	5
i1	52.804	0.95	5200	739.989	700	.5	.25	5	.5	5
i1	53.766	1.908	5200	698.456	700	.5	.25	5	.5	5
i1	55.686	0.948	5300	554.365	700	.5	.25	5	.5	5
i1	56.634	0.96	5300	523.251	700	.5	.25	5	.5	5
i1	57.602	0.952	5300	523.251	700	.5	.25	5	.5	5
i1	58.556	0.958	5400	554.365	700	.5	.25	5	.5	5
i1	59.524	1.91	5500	698.456	700	.5	.25	5	.5	5
i1	61.446	0.948	5600	739.989	700	.5	.25	5	.5	5
i1	62.404	0.95	5600	830.609	700	.5	.25	5	.5	5
i1	63.366	1.908	5700	932.328	700	.5	.25	5	.5	5
i1	65.28	1.914	5800	587.33	700	.5	.25	5	.5	5
i1	67.196	0.958	5900	523.251	700	.5	.25	5	.5	5
i1	68.156	0.958	6000	466.164	700	.5	.25	5	.5	5
i1	69.124	1.91	6100	440	700	.5	.25	5	.5	5
i1	71.034	0.96	6200	349.228	700	.5	.25	5	.5	5
i1	71.994	0.96	6300	329.628	700	.5	.25	5	.5	5
i1	72.96	0.954	6300	329.628	700	.5	.25	5	.5	5
i1	73.92	0.954	6300	349.228	700	.5	.25	5	.5	5
i1	74.882	1.912	6300	440	700	.5	.25	5	.5	5
i1	76.794	0.96	6500	466.164	700	.5	.25	5	.5	5
i1	77.766	0.948	6500	523.251	700	.5	.25	5	.5	5
i1	78.722	1.912	6600	587.33	700	.5	.25	5	.5	5
i1	80.636	1.918	6700	659.255	700	.5	.25	5	.5	5
i1	82.558	0.956	6700	587.33	700	.5	.25	5	.5	5
i1	83.514	0.96	6800	523.251	700	.5	.25	5	.5	5
i1	84.482	1.912	6800	493.883	700	.5	.25	5	.5	5
i1	86.394	0.96	7000	391.955	700	.5	.25	5	.5	5
i1	87.354	0.96	7000	369.994	700	.5	.25	5	.5	5
i1	88.322	0.952	7100	369.994	700	.5	.25	5	.5	5
i1	89.274	0.96	7200	391.955	700	.5	.25	5	.5	5
i1	90.242	1.912	7200	493.883	700	.5	.25	5	.5	5
i1	92.156	0.958	7300	523.251	700	.5	.25	5	.5	5
i1	93.12	0.954	7300	587.33	700	.5	.25	5	.5	5
i1	94.074	1.92	7500	659.255	700	.5	.25	5	.5	5
i1	95.994	1.92	7400	622.254	700	.5	.25	5	.5	5
i1	97.918	0.956	7300	554.365	700	.5	.25	5	.5	5
i1	98.88	0.954	7300	493.883	700	.5	.25	5	.5	5
i1	99.836	1.918	7200	466.164	700	.5	.25	5	.5	5
i1	101.76	0.954	7100	369.994	700	.5	.25	5	.5	5
i1	102.716	0.958	7000	349.228	700	.5	.25	5	.5	5
i1	103.676	0.958	7000	349.228	700	.5	.25	5	.5	5
i1	104.644	0.95	6900	369.994	700	.5	.25	5	.5	5
i1	105.602	1.912	6800	466.164	700	.5	.25	5	.5	5
i1	107.526	0.948	6700	493.883	700	.5	.25	5	.5	5
i1	108.48	0.954	6700	554.365	700	.5	.25	5	.5	5
i1	109.446	1.908	6700	622.254	700	.5	.25	5	.5	5

i1	111.364	1.91	6600	698.456	700	.5	.25	5	.5	5
i1	113.28	0.954	6500	622.254	700	.5	.25	5	.5	5
i1	114.246	0.948	6500	554.365	700	.5	.25	5	.5	5
i1	115.206	1.908	6400	523.251	700	.5	.25	5	.5	5
i1	117.124	0.95	6300	415.305	700	.5	.25	5	.5	5
i1	118.086	0.948	6300	391.955	700	.5	.25	5	.5	5
i1	119.04	0.954	6300	391.955	700	.5	.25	5	.5	5
i1	120.006	0.948	6200	415.305	700	.5	.25	5	.5	5
i1	120.966	1.908	6100	523.251	700	.5	.25	5	.5	5
i1	122.874	0.96	6000	554.365	700	.5	.25	5	.5	5
i1	123.838	0.956	5900	622.254	700	.5	.25	5	.5	5
i1	124.804	1.91	5900	698.456	700	.5	.25	5	.5	5
i1	126.726	1.908	5700	987.767	700	.5	.25	5	.5	5
i1	128.634	0.96	5700	880	700	.5	.25	5	.5	5
i1	129.6	0.954	5600	783.991	700	.5	.25	5	.5	5
i1	130.554	1.92	5600	739.989	700	.5	.25	5	.5	5
i1	132.482	0.952	5500	587.33	700	.5	.25	5	.5	5
i1	133.438	0.956	5400	554.365	700	.5	.25	5	.5	5
i1	134.4	0.954	5300	554.365	700	.5	.25	5	.5	5
i1	135.366	0.948	5300	587.33	700	.5	.25	5	.5	5
i1	136.324	1.91	5300	739.989	700	.5	.25	5	.5	5
i1	138.24	0.954	5200	783.991	700	.5	.25	5	.5	5
i1	139.196	0.958	5200	880	700	.5	.25	5	.5	5
i1	140.156	1.918	5100	987.767	700	.5	.25	5	.5	5
i1	142.084	1.91	5000	108.731	700	.5	.25	5	.5	5
i1	143.996	0.958	5000	987.767	700	.5	.25	5	.5	5
i1	144.962	0.952	4900	880	700	.5	.25	5	.5	5
i1	145.924	1.91	4800	830.609	700	.5	.25	5	.5	5
i1	147.844	0.95	4700	659.255	700	.5	.25	5	.5	5
i1	148.794	0.96	4700	622.254	700	.5	.25	5	.5	5
i1	149.766	0.948	4700	622.254	700	.5	.25	5	.5	5
i1	150.716	0.958	4600	659.255	700	.5	.25	5	.5	5
i1	151.682	1.912	4500	830.609	700	.5	.25	5	.5	5
i1	153.602	0.952	4400	880	700	.5	.25	5	.5	5
i1	154.558	0.956	4300	987.767	700	.5	.25	5	.5	5
i1	155.524	1.91	4300	108.731	700	.5	.25	5	.5	5
i1	157.442	1.912	4200	1046.502	700	.5	.25	5	.5	5
i1	159.358	0.956	4100	932.328	700	.5	.25	5	.5	5
i1	160.324	0.95	4100	830.609	700	.5	.25	5	.5	5
i1	161.286	1.908	4000	783.991	700	.5	.25	5	.5	5
i1	163.204	0.95	3900	622.254	700	.5	.25	5	.5	5
i1	164.164	0.95	3900	587.33	700	.5	.25	5	.5	5
i1	165.116	0.958	3800	587.33	700	.5	.25	5	.5	5
i1	166.076	0.958	3800	622.254	700	.5	.25	5	.5	5
i1	167.034	1.92	3700	783.991	700	.5	.25	5	.5	5
i1	168.956	0.958	3600	830.609	700	.5	.25	5	.5	5
i1	169.916	0.958	3600	932.328	700	.5	.25	5	.5	5
i1	170.884	1.91	3600	1046.502	700	.5	.25	5	.5	5
i1	172.804	1.91	3400	739.989	700	.5	.25	5	.5	5
i1	174.714	0.96	3300	659.255	700	.5	.25	5	.5	5
i1	175.68	0.954	3200	587.33	700	.5	.25	5	.5	5

i1	176.64	1.914	3200	554.365	700	.5	.25	5	.5	5
i1	178.558	0.956	3100	440	700	.5	.25	5	.5	5
i1	179.522	0.952	3100	415.305	700	.5	.25	5	.5	5
i1	180.476	0.958	3000	415.305	700	.5	.25	5	.5	5
i1	181.44	0.954	3000	440	700	.5	.25	5	.5	5
i1	182.394	1.92	2900	554.365	700	.5	.25	5	.5	5
i1	184.324	0.95	2800	587.33	700	.5	.25	5	.5	5
i1	185.276	0.958	2700	659.255	700	.5	.25	5	.5	5
i1	186.246	5.748	2600	739.989	700	.5	.25	5	.5	5

;**ALTO**

;Inst	Start	Dur	Amp	Freq	P6	Bal	P8	P9	Rvb	EQ
i1	3.84	0.954	2500	493.883	700	.5	.25	5	.5	5
i1	4.794	0.96	2500	523.251	700	.5	.25	5	.5	5
i1	5.76	0.954	2600	554.365	700	.5	.25	5	.5	5
i1	6.722	3.832	2600	415.305	700	.5	.25	5	.5	5
i1	12.474	3.84	2800	415.305	700	.5	.25	5	.5	5
i1	16.318	0.956	3100	554.365	700	.5	.25	5	.5	5
i1	17.278	0.956	3100	523.251	700	.5	.25	5	.5	5
i1	18.236	0.958	3100	493.883	700	.5	.25	5	.5	5
i1	19.2	0.954	3200	554.365	700	.5	.25	5	.5	5
i1	20.158	0.956	3300	587.33	700	.5	.25	5	.5	5
i1	21.116	0.958	3400	622.254	700	.5	.25	5	.5	5
i1	22.086	3.828	3400	466.164	700	.5	.25	5	.5	5
i1	27.84	3.834	3700	466.164	700	.5	.25	5	.5	5
i1	31.68	0.954	3900	622.254	700	.5	.25	5	.5	5
i1	32.64	0.954	3900	587.33	700	.5	.25	5	.5	5
i1	33.604	0.95	4000	554.365	700	.5	.25	5	.5	5
i1	34.56	0.954	4100	523.251	700	.5	.25	5	.5	5
i1	35.518	0.956	4100	554.365	700	.5	.25	5	.5	5
i1	36.476	0.958	4200	587.33	700	.5	.25	5	.5	5
i1	37.44	3.834	4200	440	700	.5	.25	5	.5	5
i1	43.2	3.834	4600	440	700	.5	.25	5	.5	5
i1	47.038	0.956	4800	587.33	700	.5	.25	5	.5	5
i1	47.994	0.96	4800	554.365	700	.5	.25	5	.5	5
i1	48.962	0.952	5000	523.251	700	.5	.25	5	.5	5
i1	49.92	0.954	5000	587.33	700	.5	.25	5	.5	5
i1	50.876	0.958	5100	622.254	700	.5	.25	5	.5	5
i1	51.846	0.948	5100	659.255	700	.5	.25	5	.5	5
i1	52.798	3.836	5200	493.883	700	.5	.25	5	.5	5
i1	58.564	3.83	5400	493.883	700	.5	.25	5	.5	5
i1	62.396	0.958	5600	659.255	700	.5	.25	5	.5	5
i1	63.354	0.96	5700	622.254	700	.5	.25	5	.5	5
i1	64.322	0.952	5700	587.33	700	.5	.25	5	.5	5
i1	65.282	0.952	5800	369.994	700	.5	.25	5	.5	5
i1	66.242	0.952	5900	391.955	700	.5	.25	5	.5	5
i1	67.206	0.948	5900	415.305	700	.5	.25	5	.5	5
i1	68.166	3.828	6000	311.127	700	.5	.25	5	.5	5
i1	73.926	3.828	6300	311.127	700	.5	.25	5	.5	5
i1	77.756	0.958	6500	415.305	700	.5	.25	5	.5	5
i1	78.72	0.954	6600	391.955	700	.5	.25	5	.5	5

i1	79.674	0.96	6600	369.994	700	.5	.25	5	.5	5
i1	80.636	0.958	6700	415.305	700	.5	.25	5	.5	5
i1	81.6	0.954	6700	440	700	.5	.25	5	.5	5
i1	82.562	0.952	6700	466.164	700	.5	.25	5	.5	5
i1	83.516	3.838	6800	349.228	700	.5	.25	5	.5	5
i1	89.284	3.83	7200	349.228	700	.5	.25	5	.5	5
i1	93.126	0.948	7300	466.164	700	.5	.25	5	.5	5
i1	94.086	0.948	7500	440	700	.5	.25	5	.5	5
i1	95.036	0.958	7500	415.305	700	.5	.25	5	.5	5
i1	95.996	0.958	7400	391.955	700	.5	.25	5	.5	5
i1	96.966	0.948	7300	415.305	700	.5	.25	5	.5	5
i1	97.924	0.95	7300	440	700	.5	.25	5	.5	5
i1	98.886	3.828	7300	329.628	700	.5	.25	5	.5	5
i1	104.644	3.83	6900	329.628	700	.5	.25	5	.5	5
i1	108.482	0.952	6700	440	700	.5	.25	5	.5	5
i1	109.444	0.95	6700	415.305	700	.5	.25	5	.5	5
i1	110.398	0.956	6600	391.955	700	.5	.25	5	.5	5
i1	111.356	0.958	6600	440	700	.5	.25	5	.5	5
i1	112.326	0.948	6600	466.164	700	.5	.25	5	.5	5
i1	113.286	0.948	6500	493.883	700	.5	.25	5	.5	5
i1	114.238	3.836	6500	369.994	700	.5	.25	5	.5	5
i1	120.006	3.828	6200	369.994	700	.5	.25	5	.5	5
i1	123.844	0.95	5900	493.883	700	.5	.25	5	.5	5
i1	124.806	0.948	5900	466.164	700	.5	.25	5	.5	5
i1	125.758	0.956	5800	440	700	.5	.25	5	.5	5
i1	126.714	0.96	5700	622.254	700	.5	.25	5	.5	5
i1	127.682	0.952	5700	659.255	700	.5	.25	5	.5	5
i1	128.642	0.952	5700	698.456	700	.5	.25	5	.5	5
i1	129.596	3.838	5600	523.251	700	.5	.25	5	.5	5
i1	135.364	3.83	5300	523.251	700	.5	.25	5	.5	5
i1	139.204	0.95	5200	698.456	700	.5	.25	5	.5	5
i1	140.164	0.95	5100	659.255	700	.5	.25	5	.5	5
i1	141.12	0.954	5100	622.254	700	.5	.25	5	.5	5
i1	142.084	0.95	5000	698.456	700	.5	.25	5	.5	5
i1	143.036	0.958	5000	739.989	700	.5	.25	5	.5	5
i1	144.006	0.948	5000	783.991	700	.5	.25	5	.5	5
i1	144.962	3.832	4900	587.33	700	.5	.25	5	.5	5
i1	150.718	3.836	4600	587.33	700	.5	.25	5	.5	5
i1	154.562	0.952	4300	783.991	700	.5	.25	5	.5	5
i1	155.52	0.954	4300	739.989	700	.5	.25	5	.5	5
i1	156.476	0.958	4200	698.456	700	.5	.25	5	.5	5
i1	157.438	0.956	4200	659.255	700	.5	.25	5	.5	5
i1	158.406	0.948	4100	698.456	700	.5	.25	5	.5	5
i1	159.364	0.95	4100	739.989	700	.5	.25	5	.5	5
i1	160.314	3.84	4100	554.365	700	.5	.25	5	.5	5
i1	166.08	3.834	3800	554.365	700	.5	.25	5	.5	5
i1	169.916	0.958	3600	739.989	700	.5	.25	5	.5	5
i1	170.88	0.954	3600	698.456	700	.5	.25	5	.5	5
i1	171.838	0.956	3500	659.255	700	.5	.25	5	.5	5
i1	172.8	0.954	3400	466.164	700	.5	.25	5	.5	5
i1	173.766	0.948	3400	493.883	700	.5	.25	5	.5	5

```
i1      174.718     0.956   3300    523.251700      .5      .25     5       .5      5
i1      175.682     3.832   3200    391.955700      .5      .25     5       .5      5
i1      181.44      3.834   3000    391.955700      .5      .25     5       .5      5
i1      185.284     0.95    2700    523.251700      .5      .25     5       .5      5
i1      186.234     0.96    2600    493.883700      .5      .25     5       .5      5
i1      187.2       4.794   2600    466.164700      .5      .25     5       .5      5

;TENOR
;Inst   St          Dur     Amp     Freq
i2      3.838       0.956   2500    293.665         .5      .5      .25     5       .3      5
i2      4.806       1.908   2500    329.628         .5      .5      .       5       .       .
i2      6.716       0.958   2600    233.082         .5      .5      .       5       .       .
i2      7.686       0.948   2600    246.942         .5      .5      .       5       .       .
i2      8.634       0.96    2600    261.626         .5      .5      .       5       .       .
i2      9.594       0.96    2600    277.183         .5      .5      .       5       .       .
i2      10.554      0.96    2700    311.127         .5      .5      .       5       .       .
i2      11.514      0.96    2800    311.127         .5      .5      .       5       .       .
i2      12.476      0.958   2800    277.183         .5      .5      .       5       .       .
i2      13.434      0.96    2900    261.626         .5      .5      .       5       .       .
i2      14.396      0.958   3000    246.942         .5      .5      .       5       .       .
i2      15.364      0.95    3000    233.082         .5      .5      .       5       .       .
i2      16.318      1.916   3100    329.628         .5      .5      .       5       .       .
i2      18.234      0.96    3100    293.665         .5      .5      .       5       .       .
i2      19.196      0.958   3200    329.628         .5      .5      .       5       .       .
i2      20.154      1.92    3300    369.994         .5      .5      .       5       .       .
i2      22.076      0.958   3400    261.626         .5      .5      .       5       .       .
i2      23.04       0.954   3500    277.183         .5      .5      .       5       .       .
i2      23.994      0.96    3600    293.665         .5      .5      .       5       .       .
i2      24.96       0.954   3600    311.127         .5      .5      .       5       .       .
i2      25.916      0.958   3600    349.228         .5      .5      .       5       .       .
i2      26.882      0.952   3700    349.228         .5      .5      .       5       .       .
i2      27.844      0.95    3700    311.127         .5      .5      .       5       .       .
i2      28.804      0.95    3800    293.665         .5      .5      .       5       .       .
i2      29.76       0.954   3800    277.183         .5      .5      .       5       .       .
i2      30.714      0.96    3900    261.626         .5      .5      .       5       .       .
i2      31.68       1.914   3900    369.994         .5      .5      .       5       .       .
i2      33.596      0.958   4000    329.628         .5      .5      .       5       .       .
i2      34.56       0.954   4100    311.127         .5      .5      .       5       .       .
i2      35.514      1.92    4100    349.228         .5      .5      .       5       .       .
i2      37.444      0.95    4200    246.942         .5      .5      .       5       .       .
i2      38.4        0.954   4300    261.626         .5      .5      .       5       .       .
i2      39.36       0.954   4300    277.183         .5      .5      .       5       .       .
i2      40.322      0.952   4400    293.665         .5      .5      .       5       .       .
i2      41.284      0.95    4500    329.628         .5      .5      .       5       .       .
i2      42.246      0.948   4500    329.628         .5      .5      .       5       .       .
i2      43.2        0.954   4600    293.665         .5      .5      .       5       .       .
i2      44.154      0.96    4700    277.183         .5      .5      .       5       .       .
i2      45.12       0.954   4700    261.626         .5      .5      .       5       .       .
i2      46.082      0.952   4700    246.942         .5      .5      .       5       .       .
i2      47.04       1.914   4800    349.228         .5      .5      .       5       .       .
i2      48.96       0.954   5000    311.127         .5      .5      .       5       .       .
```

i2	49.926	0.948	5000	349.228	.5	.5	.	5	.	.
i2	50.88	1.914	5100	391.955	.5	.5	.	5	.	.
i2	52.796	0.958	5200	277.183	.5	.5	.	5	.	.
i2	53.756	0.958	5200	293.665	.5	.5	.	5	.	.
i2	54.716	0.958	5200	311.127	.5	.5	.	5	.	.
i2	55.676	0.958	5300	329.628	.5	.5	.	5	.	.
i2	56.638	0.956	5300	369.994	.5	.5	.	5	.	.
i2	57.596	0.958	5300	369.994	.5	.5	.	5	.	.
i2	58.566	0.948	5400	329.628	.5	.5	.	5	.	.
i2	59.514	0.96	5500	311.127	.5	.5	.	5	.	.
i2	60.482	0.952	5500	293.665	.5	.5	.	5	.	.
i2	61.436	0.958	5600	277.183	.5	.5	.	5	.	.
i2	62.4	1.914	5600	391.955	.5	.5	.	5	.	.
i2	64.316	0.958	5700	349.228	.5	.5	.	5	.	.
i2	65.276	0.958	5800	220	.5	.5	.	5	.	.
i2	66.246	1.908	5900	246.942	.5	.5	.	5	.	.
i2	68.166	0.948	6000	174.614	.5	.5	.	5	.	.
i2	69.116	0.958	6100	184.997	.5	.5	.	5	.	.
i2	70.076	0.958	6100	195.998	.5	.5	.	5	.	.
i2	71.042	0.952	6200	207.654	.5	.5	.	5	.	.
i2	72.006	0.948	6300	233.082	.5	.5	.	5	.	.
i2	72.954	0.96	6300	233.082	.5	.5	.	5	.	.
i2	73.918	0.956	6300	207.654	.5	.5	.	5	.	.
i2	74.884	0.95	6300	195.998	.5	.5	.	5	.	.
i2	75.84	0.954	6400	184.997	.5	.5	.	5	.	.
i2	76.794	0.96	6500	174.614	.5	.5	.	5	.	.
i2	77.756	1.918	6500	246.942	.5	.5	.	5	.	.
i2	79.678	0.956	6600	220	.5	.5	.	5	.	.
i2	80.644	0.95	6700	246.942	.5	.5	.	5	.	.
i2	81.604	1.91	6700	277.183	.5	.5	.	5	.	.
i2	83.522	0.952	6800	195.998	.5	.5	.	5	.	.
i2	84.474	0.96	6800	207.654	.5	.5	.	5	.	.
i2	85.434	0.96	6900	220	.5	.5	.	5	.	.
i2	86.394	0.96	7000	233.082	.5	.5	.	5	.	.
i2	87.364	0.95	7000	261.626	.5	.5	.	5	.	.
i2	88.322	0.952	7100	261.626	.5	.5	.	5	.	.
i2	89.286	0.948	7200	233.082	.5	.5	.	5	.	.
i2	90.24	0.954	7200	220	.5	.5	.	5	.	.
i2	91.2	0.954	7300	207.654	.5	.5	.	5	.	.
i2	92.154	0.96	7300	195.998	.5	.5	.	5	.	.
i2	93.126	1.908	7300	277.183	.5	.5	.	5	.	.
i2	95.04	0.954	7500	246.942	.5	.5	.	5	.	.
i2	95.996	0.958	7400	233.082	.5	.5	.	5	.	.
i2	96.966	1.908	7300	261.626	.5	.5	.	5	.	.
i2	98.88	0.954	7300	184.997	.5	.5	.	5	.	.
i2	99.84	0.954	7200	195.998	.5	.5	.	5	.	.
i2	100.8	0.954	7200	207.654	.5	.5	.	5	.	.
i2	101.758	0.956	7100	220	.5	.5	.	5	.	.
i2	102.726	0.948	7000	246.942	.5	.5	.	5	.	.
i2	103.686	0.948	7000	246.942	.5	.5	.	5	.	.
i2	104.646	0.948	6900	220	.5	.5	.	5	.	.

i2	105.602	0.952	6800	207.654	.5	.5	.	5	.	.
i2	106.562	0.952	6800	195.998	.5	.5	.	5	.	.
i2	107.522	0.952	6700	184.997	.5	.5	.	5	.	.
i2	108.474	1.92	6700	261.626	.5	.5	.	5	.	.
i2	110.404	0.95	6600	233.082	.5	.5	.	5	.	.
i2	111.36	0.954	6600	261.626	.5	.5	.	5	.	.
i2	112.318	1.916	6600	293.665	.5	.5	.	5	.	.
i2	114.246	0.948	6500	207.654	.5	.5	.	5	.	.
i2	115.206	0.948	6400	220	.5	.5	.	5	.	.
i2	116.154	0.96	6300	233.082	.5	.5	.	5	.	.
i2	117.116	0.958	6300	246.942	.5	.5	.	5	.	.
i2	118.074	0.96	6300	277.183	.5	.5	.	5	.	.
i2	119.038	0.956	6300	277.183	.5	.5	.	5	.	.
i2	119.998	0.956	6200	246.942	.5	.5	.	5	.	.
i2	120.964	0.95	6100	233.082	.5	.5	.	5	.	.
i2	121.922	0.952	6100	220	.5	.5	.	5	.	.
i2	122.876	0.958	6000	207.654	.5	.5	.	5	.	.
i2	123.84	1.914	5900	293.665	.5	.5	.	5	.	.
i2	125.756	0.958	5800	261.626	.5	.5	.	5	.	.
i2	126.714	0.96	5700	369.994	.5	.5	.	5	.	.
i2	127.68	1.914	5700	415.305	.5	.5	.	5	.	.
i2	129.602	0.952	5600	293.665	.5	.5	.	5	.	.
i2	130.558	0.956	5600	311.127	.5	.5	.	5	.	.
i2	131.524	0.95	5500	329.628	.5	.5	.	5	.	.
i2	132.48	0.954	5500	349.228	.5	.5	.	5	.	.
i2	133.442	0.952	5400	391.955	.5	.5	.	5	.	.
i2	134.404	0.95	5300	391.955	.5	.5	.	5	.	.
i2	135.356	0.958	5300	349.228	.5	.5	.	5	.	.
i2	136.32	0.954	5300	329.628	.5	.5	.	5	.	.
i2	137.278	0.956	5200	311.127	.5	.5	.	5	.	.
i2	138.236	0.958	5200	293.665	.5	.5	.	5	.	.
i2	139.2	1.914	5200	415.305	.5	.5	.	5	.	.
i2	141.126	0.948	5100	369.994	.5	.5	.	5	.	.
i2	142.078	0.956	5000	415.305	.5	.5	.	5	.	.
i2	143.042	1.912	5000	466.164	.5	.5	.	5	.	.
i2	144.962	0.952	4900	329.628	.5	.5	.	5	.	.
i2	145.922	0.952	4800	349.228	.5	.5	.	5	.	.
i2	146.874	0.96	4800	369.994	.5	.5	.	5	.	.
i2	147.846	0.948	4700	391.955	.5	.5	.	5	.	.
i2	148.806	0.948	4700	440	.5	.5	.	5	.	.
i2	149.762	0.952	4700	440	.5	.5	.	5	.	.
i2	150.718	0.956	4600	391.955	.5	.5	.	5	.	.
i2	151.68	0.954	4500	369.994	.5	.5	.	5	.	.
i2	152.642	0.952	4500	349.228	.5	.5	.	5	.	.
i2	153.594	0.96	4400	329.628	.5	.5	.	5	.	.
i2	154.558	1.916	4300	466.164	.5	.5	.	5	.	.
i2	156.48	0.954	4200	415.305	.5	.5	.	5	.	.
i2	157.444	0.95	4200	391.955	.5	.5	.	5	.	.
i2	158.398	1.916	4100	440	.5	.5	.	5	.	.
i2	160.316	0.958	4100	311.127	.5	.5	.	5	.	.
i2	161.282	0.952	4000	329.628	.5	.5	.	5	.	.

```
i2      162.234     0.96    3900    349.228     .5      .5      .       5       .       .
i2      163.202     0.952   3900    369.994     .5      .5      .       5       .       .
i2      164.156     0.958   3900    415.305     .5      .5      .       5       .       .
i2      165.114     0.96    3800    415.305     .5      .5      .       5       .       .
i2      166.076     0.958   3800    369.994     .5      .5      .       5       .       .
i2      167.036     0.958   3700    349.228     .5      .5      .       5       .       .
i2      168         0.954   3700    329.628     .5      .5      .       5       .       .
i2      168.962     0.952   3600    311.127     .5      .5      .       5       .       .
i2      169.914     1.92    3600    440         .5      .5      .       5       .       .
i2      171.834     0.96    3500    391.955     .5      .5      .       5       .       .
i2      172.796     0.958   3400    277.183     .5      .5      .       5       .       .
i2      173.756     1.918   3400    311.127     .5      .5      .       5       .       .
i2      175.68      0.954   3200    220         .5      .5      .       5       .       .
i2      176.636     0.958   3200    233.082     .5      .5      .       5       .       .
i2      177.596     0.958   3100    246.942     .5      .5      .       5       .       .
i2      178.564     0.95    3100    261.626     .5      .5      .       5       .       .
i2      179.516     0.958   3100    293.665     .5      .5      .       5       .       .
i2      180.482     0.952   3000    293.665     .5      .5      .       5       .       .
i2      181.434     0.96    3000    261.626     .5      .5      .       5       .       .
i2      182.398     0.956   2900    246.942     .5      .5      .       5       .       .
i2      183.36      0.954   2800    233.082     .5      .5      .       5       .       .
i2      184.324     0.95    2800    220         .5      .5      .       5       .       .
i2      185.284     1.91    2700    311.127     .5      .5      .       5       .       .
i2      187.204     4.79    2600    277.183     .5      .5      .       5       .       .

;BASS
;Inst   St          Dur     Amp     Freq
i2      4.804       0.95    2500    184.997     .3      .3      .3      5       5       5
i2      5.758       0.956   2600    110         .3      .3      .       5       .       .
i2      7.678       0.956   2600    97.999      .3      .3      .       5       .       .
i2      8.638       1.916   2600    164.813     .3      .3      .       5       .       .
i2      10.558      0.476   2700    184.997     .3      .3      .       5       .       .
i2      11.044      0.47    2700    174.614     .3      .3      .       5       .       .
i2      11.52       0.474   2800    174.614     .3      .3      .       5       .       .
i2      11.998      0.476   2800    184.997     .3      .3      .       5       .       .
i2      12.474      1.92    2800    164.813     .3      .3      .       5       .       .
i2      14.398      0.956   3000    97.999      .3      .3      .       5       .       .
i2      16.314      0.96    3100    110         .3      .3      .       5       .       .
i2      17.274      0.96    3100    184.997     .3      .3      .       5       .       .
i2      20.158      0.956   3300    207.654     .3      .3      .       5       .       .
i2      21.122      0.952   3400    123.471     .3      .3      .       5       .       .
i2      23.046      0.948   3500    110         .3      .3      .       5       .       .
i2      24          1.914   3600    184.997     .3      .3      .       5       .       .
i2      25.916      0.478   3600    207.654     .3      .3      .       5       .       .
i2      26.406      0.468   3700    195.998     .3      .3      .       5       .       .
i2      26.882      0.472   3700    195.998     .3      .3      .       5       .       .
i2      27.356      0.478   3700    207.654     .3      .3      .       5       .       .
i2      27.838      1.916   3700    184.997     .3      .3      .       5       .       .
i2      29.756      0.958   3800    110         .3      .3      .       5       .       .
i2      31.674      0.96    3900    123.471     .3      .3      .       5       .       .
i2      32.634      0.96    3900    207.654     .3      .3      .       5       .       .
```

i2	35.522	0.952	4100	195.998	.3	.3	.	5	.	.
i2	36.486	0.948	4200	116.541	.3	.3	.	5	.	.
i2	38.404	0.95	4300	103.826	.3	.3	.	5	.	.
i2	39.366	1.908	4300	174.614	.3	.3	.	5	.	.
i2	41.282	0.472	4500	195.998	.3	.3	.	5	.	.
i2	41.762	0.472	4500	184.997	.3	.3	.	5	.	.
i2	42.236	0.478	4500	184.997	.3	.3	.	5	.	.
i2	42.716	0.478	4600	195.998	.3	.3	.	5	.	.
i2	43.196	1.918	4600	174.614	.3	.3	.	5	.	.
i2	45.12	0.954	4700	103.826	.3	.3	.	5	.	.
i2	47.042	0.952	4800	116.541	.3	.3	.	5	.	.
i2	47.994	0.96	4800	195.998	.3	.3	.	5	.	.
i2	50.878	0.956	5100	220	.3	.3	.	5	.	.
i2	51.84	0.954	5100	130.813	.3	.3	.	5	.	.
i2	53.766	0.948	5200	116.541	.3	.3	.	5	.	.
i2	54.726	1.908	5200	195.998	.3	.3	.	5	.	.
i2	56.642	0.472	5300	220	.3	.3	.	5	.	.
i2	57.12	0.474	5300	207.654	.3	.3	.	5	.	.
i2	57.596	0.478	5300	207.654	.3	.3	.	5	.	.
i2	58.086	0.468	5400	220	.3	.3	.	5	.	.
i2	58.558	1.916	5400	195.998	.3	.3	.	5	.	.
i2	60.486	0.948	5500	116.541	.3	.3	.	5	.	.
i2	62.394	0.96	5600	130.813	.3	.3	.	5	.	.
i2	63.358	0.956	5700	220	.3	.3	.	5	.	.
i2	66.236	0.958	5900	138.591	.3	.3	.	5	.	.
i2	67.2	0.954	5900	82.407	.3	.3	.	5	.	.
i2	69.126	0.948	6100	73.416	.3	.3	.	5	.	.
i2	70.084	1.91	6100	123.471	.3	.3	.	5	.	.
i2	71.994	0.48	6300	138.591	.3	.3	.	5	.	.
i2	72.48	0.474	6300	130.813	.3	.3	.	5	.	.
i2	72.966	0.468	6300	130.813	.3	.3	.	5	.	.
i2	73.446	0.468	6300	138.591	.3	.3	.	5	.	.
i2	73.916	1.918	6300	123.471	.3	.3	.	5	.	.
i2	75.836	0.958	6400	73.416	.3	.3	.	5	.	.
i2	77.756	0.958	6500	82.407	.3	.3	.	5	.	.
i2	78.72	0.954	6600	138.591	.3	.3	.	5	.	.
i2	81.598	0.956	6700	155.563	.3	.3	.	5	.	.
i2	82.56	0.954	6700	92.499	.3	.3	.	5	.	.
i2	84.476	0.958	6800	82.407	.3	.3	.	5	.	.
i2	85.436	1.918	6900	138.591	.3	.3	.	5	.	.
i2	87.356	0.478	7000	155.563	.3	.3	.	5	.	.
i2	87.846	0.468	7100	146.832	.3	.3	.	5	.	.
i2	88.322	0.472	7100	146.832	.3	.3	.	5	.	.
i2	88.802	0.472	7200	155.563	.3	.3	.	5	.	.
i2	89.284	1.91	7200	138.591	.3	.3	.	5	.	.
i2	91.204	0.95	7300	82.407	.3	.3	.	5	.	.
i2	93.116	0.958	7300	92.499	.3	.3	.	5	.	.
i2	94.074	0.96	7500	155.563	.3	.3	.	5	.	.
i2	96.96	0.954	7300	146.832	.3	.3	.	5	.	.
i2	97.916	0.958	7300	87.307	.3	.3	.	5	.	.
i2	99.834	0.96	7200	77.782	.3	.3	.	5	.	.

i2	100.802	1.912	7200	130.813	.3	.3	.	5	.	.
i2	102.722	0.472	7000	146.832	.3	.3	.	5	.	.
i2	103.204	0.47	7000	138.591	.3	.3	.	5	.	.
i2	103.676	0.478	7000	138.591	.3	.3	.	5	.	.
i2	104.166	0.468	6900	146.832	.3	.3	.	5	.	.
i2	104.634	1.92	6900	130.813	.3	.3	.	5	.	.
i2	106.56	0.954	6800	77.782	.3	.3	.	5	.	.
i2	108.484	0.95	6700	87.307	.3	.3	.	5	.	.
i2	109.442	0.952	6700	146.832	.3	.3	.	5	.	.
i2	112.318	0.956	6600	164.813	.3	.3	.	5	.	.
i2	113.286	0.948	6500	97.999	.3	.3	.	5	.	.
i2	115.2	0.954	6400	87.307	.3	.3	.	5	.	.
i2	116.154	1.92	6300	146.832	.3	.3	.	5	.	.
i2	118.074	0.48	6300	164.813	.3	.3	.	5	.	.
i2	118.556	0.478	6300	155.563	.3	.3	.	5	.	.
i2	119.042	0.472	6300	155.563	.3	.3	.	5	.	.
i2	119.522	0.472	6300	164.813	.3	.3	.	5	.	.
i2	120.002	1.912	6200	146.83	.3	.3	.	5	.	.
i2	121.916	0.958	6100	87.307	.3	.3	.	5	.	.
i2	123.838	0.956	5900	97.999	.3	.3	.	5	.	.
i2	124.8	0.954	5900	164.813	.3	.3	.	5	.	.
i2	127.684	0.95	5700	233.082	.3	.3	.	5	.	.
i2	128.642	0.952	5700	138.591	.3	.3	.	5	.	.
i2	130.562	0.952	5600	123.471	.3	.3	.	5	.	.
i2	131.518	1.916	5500	207.654	.3	.3	.	5	.	.
i2	133.438	0.476	5400	233.082	.3	.3	.	5	.	.
i2	133.92	0.474	5400	220	.3	.3	.	5	.	.
i2	134.404	0.47	5300	220	.3	.3	.	5	.	.
i2	134.884	0.47	5300	233.082	.3	.3	.	5	.	.
i2	135.354	1.92	5300	207.654	.3	.3	.	5	.	.
i2	137.286	0.948	5200	123.471	.3	.3	.	5	.	.
i2	139.206	0.948	5200	138.591	.3	.3	.	5	.	.
i2	140.158	0.956	5100	233.082	.3	.3	.	5	.	.
i2	143.04	0.954	5000	261.626	.3	.3	.	5	.	.
i2	144.002	0.952	5000	155.563	.3	.3	.	5	.	.
i2	145.92	0.954	4800	138.591	.3	.3	.	5	.	.
i2	146.882	1.912	4800	233.082	.3	.3	.	5	.	.
i2	148.794	0.48	4700	261.626	.3	.3	.	5	.	.
i2	149.278	0.476	4700	246.942	.3	.3	.	5	.	.
i2	149.758	0.476	4700	246.942	.3	.3	.	5	.	.
i2	150.246	0.468	4600	261.626	.3	.3	.	5	.	.
i2	150.718	1.916	4600	233.082	.3	.3	.	5	.	.
i2	152.64	0.954	4500	138.591	.3	.3	.	5	.	.
i2	154.556	0.958	4300	155.563	.3	.3	.	5	.	.
i2	155.52	0.954	4300	261.626	.3	.3	.	5	.	.
i2	158.398	0.956	4100	246.942	.3	.3	.	5	.	.
i2	159.364	0.95	4100	146.832	.3	.3	.	5	.	.
i2	161.274	0.96	4000	130.813	.3	.3	.	5	.	.
i2	162.244	1.91	3900	220	.3	.3	.	5	.	.
i2	164.16	0.474	3900	246.942	.3	.3	.	5	.	.
i2	164.642	0.472	3800	233.082	.3	.3	.	5	.	.

```
i2      165.116     0.478   3800    233.082     .3      .3      .       5       .       .
i2      165.594     0.48    3800    246.942     .3      .3      .       5       .       .
i2      166.076     1.918   3800    220         .3      .3      .       5       .       .
i2      167.996     0.958   3700    130.813     .3      .3      .       5       .       .
i2      169.92      0.954   3600    146.832     .3      .3      .       5       .       .
i2      170.886     0.948   3600    246.942     .3      .3      .       5       .       .
i2      173.762     0.952   3400    174.614     .3      .3      .       5       .       .
i2      174.724     0.95    3300    103.826     .3      .3      .       5       .       .
i2      176.634     0.96    3200    92.499      .3      .3      .       5       .       .
i2      177.598     1.916   3100    155.563     .3      .3      .       5       .       .
i2      179.516     0.478   3100    174.614     .3      .3      .       5       .       .
i2      179.994     0.48    3100    164.813     .3      .3      .       5       .       .
i2      180.486     0.468   3000    164.813     .3      .3      .       5       .       .
i2      180.96      0.474   3000    174.614     .3      .3      .       5       .       .
i2      181.446     1.908   3000    155.563     .3      .3      .       5       .       .
i2      183.366     0.948   2800    92.499      .3      .3      .       5       .       .
i2      185.274     0.96    2700    103.826     .3      .3      .       5       .       .
i2      186.234     0.96    2600    174.614     .3      .3      .       5       .       .
i2      187.198     4.796   4200    97.999      .3      .3      .       5       .       .

;Trumpet
;Inst   St          Dur     Amp     Freq        P6      P7      P8
i3      3.842       3.642   7400    391.955     .2      .3      4
i3      7.684       1.816   7200    391.955     .2      .3      4
i3      9.598       1.822   7100    369.994     .2      .3      4
i3      11.524      1.816   7100    329.628     .2      .3      4
i3      13.436      1.824   7000    293.665     .2      .3      4
i3      15.358      3.646   6900    391.955     .2      .3      4
i3      19.2        3.644   6700    440         .2      .3      4
i3      23.036      3.648   6600    493.883     .2      .3      4
i3      26.88       3.644   6400    493.883     .2      .3      4
i3      30.716      1.824   6200    493.883     .2      .3      4
i3      32.638      1.822   6100    493.883     .2      .3      4
i3      34.556      1.824   6000    440         .2      .3      4
i3      36.476      1.824   5900    391.955     .2      .3      4
i3      38.404      3.64    5900    523.251     .2      .3      4
i3      42.238      3.646   5700    493.883     .2      .3      4
i3      46.08       3.644   5500    440         .2      .3      4
i3      49.92       3.644   5400    391.955     .2      .3      4
i3      53.758      1.822   5200    440         .2      .3      4
i3      55.678      1.822   5100    493.883     .2      .3      4
i3      57.604      1.816   5000    440         .2      .3      4
i3      59.518      1.822   4900    391.955     .2      .3      4
i3      61.444      3.64    4900    329.628     .2      .3      4
i3      65.278      3.646   4700    369.994     .2      .3      4
i3      69.12       3.644   4500    391.955     .2      .3      4
i3      72.956      3.648   4300    587.33      .2      .3      4
i3      76.802      1.818   4200    493.883     .2      .3      4
i3      78.724      1.816   4100    391.955     .2      .3      4
i3      80.638      1.822   4000    440         .2      .3      4
i3      82.562      1.818   3900    523.251     .2      .3      4
```

```
i3      84.484      3.64    3800    493.883     .2      .3      4
i3      88.318      3.646   3700    440         .2      .3      4
i3      92.156      3.648   3500    391.955     .2      .3      4
i3      95.996      3.648   3400    391.955     .2      .3      4
i3      99.842      1.818   3600    391.955     .2      .3      4
i3      101.762     1.818   3700    369.994     .2      .3      4
i3      103.682     1.818   3700    329.628     .2      .3      4
i3      105.598     1.822   3800    293.665     .2      .3      4
i3      107.524     3.64    3900    391.955     .2      .3      4
i3      111.358     3.646   4100    440         .2      .3      4
i3      115.204     3.64    4200    493.883     .2      .3      4
i3      119.044     3.64    4300    493.883     .2      .3      4
i3      122.884     1.816   4500    493.883     .2      .3      4
i3      124.796     1.824   4600    493.883     .2      .3      4
i3      126.722     1.818   4700    440         .2      .3      4
i3      128.644     1.816   4700    391.955     .2      .3      4
i3      130.558     3.646   4900    523.251     .2      .3      4
i3      134.402     3.642   5000    493.883     .2      .3      4
i3      138.244     3.64    5100    440         .2      .3      4
i3      142.084     3.64    5300    391.955     .2      .3      4
i3      145.916     1.824   5500    440         .2      .3      4
i3      147.84      1.82    5500    493.883     .2      .3      4
i3      149.764     1.816   5600    440         .2      .3      4
i3      151.676     1.824   5700    391.955     .2      .3      4
i3      153.598     3.646   5800    329.628     .2      .3      4
i3      157.444     3.64    5900    369.994     .2      .3      4
i3      161.282     3.642   6100    391.955     .2      .3      4
i3      165.122     3.642   6200    587.33      .2      .3      4
i3      168.962     1.818   6400    493.883     .2      .3      4
i3      170.884     1.816   6400    391.955     .2      .3      4
i3      172.798     1.822   6600    440         .2      .3      4
i3      174.722     1.818   6600    523.251     .2      .3      4
i3      176.64      3.644   6700    493.883     .2      .3      4
i3      180.484     3.64    6800    440         .2      .3      4
i3      184.316     7.488   7000    391.955     .2      .3      4

;French Horn
;Inst   St          Dur     Amp     Freq        P6      P7      P8
i3      3.842       3.642   8300    293.665     .2      .3      4
i3      7.68        1.82    8100    293.665     .2      .3      4
i3      9.602       1.818   8000    293.665     .2      .3      4
i3      11.516      1.824   8000    261.626     .2      .3      4
i3      13.444      1.816   7800    246.942     .2      .3      4
i3      15.356      3.648   7800    246.942     .2      .3      4
i3      19.198      3.646   7600    293.665     .2      .3      4
i3      23.038      3.646   7400    293.665     .2      .3      4
i3      26.876      3.648   7200    293.665     .2      .3      4
i3      30.718      1.822   7000    293.665     .2      .3      4
i3      32.64       1.82    6900    293.665     .2      .3      4
i3      34.564      1.816   6800    293.665     .2      .3      4
i3      36.478      1.822   6700    246.942     .2      .3      4
```

i3	38.398	3.646	6600	329.628	.2	.3	4
i3	42.236	3.648	6400	293.665	.2	.3	4
i3	46.082	3.642	6200	293.665	.2	.3	4
i3	49.92	3.644	6000	391.955	.2	.3	4
i3	53.764	1.816	5800	293.665	.2	.3	4
i3	55.68	1.82	5700	293.665	.2	.3	4
i3	57.596	1.824	5600	293.665	.2	.3	4
i3	59.524	1.816	5500	293.665	.2	.3	4
i3	61.444	3.64	5500	261.626	.2	.3	4
i3	65.28	3.644	5300	261.626	.2	.3	4
i3	69.118	3.646	5100	293.665	.2	.3	4
i3	72.958	3.646	4900	293.665	.2	.3	4
i3	76.798	1.822	4700	293.665	.2	.3	4
i3	78.718	1.822	4600	391.955	.2	.3	4
i3	80.642	1.818	4500	369.994	.2	.3	4
i3	82.558	0.91	4400	329.628	.2	.3	4
i3	83.516	0.912	4400	369.994	.2	.3	4
i3	84.478	3.646	4300	391.955	.2	.3	4
i3	88.324	3.64	4100	369.994	.2	.3	4
i3	92.164	3.64	3900	246.942	.2	.3	4
i3	96	3.644	3900	293.665	.2	.3	4
i3	99.836	1.824	4000	293.665	.2	.3	4
i3	101.764	1.816	4100	293.665	.2	.3	4
i3	103.678	1.822	4200	261.626	.2	.3	4
i3	105.596	1.824	4300	246.942	.2	.3	4
i3	107.52	3.644	4400	246.942	.2	.3	4
i3	111.358	3.646	4600	293.665	.2	.3	4
i3	115.204	3.64	4800	293.665	.2	.3	4
i3	119.044	3.64	4900	293.665	.2	.3	4
i3	122.882	1.818	5100	293.665	.2	.3	4
i3	124.798	1.822	5100	293.665	.2	.3	4
i3	126.722	1.818	5300	293.665	.2	.3	4
i3	128.636	1.824	5300	246.942	.2	.3	4
i3	130.56	3.644	5500	329.628	.2	.3	4
i3	134.4	3.644	5600	293.665	.2	.3	4
i3	138.244	3.64	5800	293.665	.2	.3	4
i3	142.082	3.642	6000	391.955	.2	.3	4
i3	145.92	1.82	6200	293.665	.2	.3	4
i3	147.842	1.818	6200	293.665	.2	.3	4
i3	149.756	1.824	6300	293.665	.2	.3	4
i3	151.68	1.82	6400	293.665	.2	.3	4
i3	153.6	3.644	6500	261.626	.2	.3	4
i3	157.438	3.646	6700	261.626	.2	.3	4
i3	161.284	3.64	6900	293.665	.2	.3	4
i3	165.12	3.644	7000	293.665	.2	.3	4
i3	168.96	1.82	7200	293.665	.2	.3	4
i3	170.876	1.824	7200	391.955	.2	.3	4
i3	172.8	1.82	7400	369.994	.2	.3	4
i3	174.718	0.91	7400	329.628	.2	.3	4
i3	175.684	0.904	7500	369.994	.2	.3	4
i3	176.64	3.644	7600	391.955	.2	.3	4

```
i3      180.482     3.642   7700    369.994     .2      .3      4
i3      184.316     7.488   7900    246.942     .2      .3      4

;Trombone
;Inst   St          Dur     Amp     Freq        P6      P7      P8
i4      3.836       3.648   9100    246.942     .7      .3      4
i4      7.68        1.82    8900    246.942     .7      .3      4
i4      9.596       1.824   8800    220         .7      .3      4
i4      11.522      1.818   8700    195.998     .7      .3      4
i4      13.438      1.822   8600    184.997     .7      .3      4
i4      15.356      3.648   8500    195.998     .7      .3      4
i4      19.202      3.642   8300    184.997     .7      .3      4
i4      23.042      3.642   8100    195.998     .7      .3      4
i4      26.878      3.646   7900    195.998     .7      .3      4
i4      30.716      1.824   7700    195.998     .7      .3      4
i4      32.64       1.82    7500    195.998     .7      .3      4
i4      34.56       1.82    7400    184.997     .7      .3      4
i4      36.476      1.824   7300    195.998     .7      .3      4
i4      38.396      3.648   7200    195.998     .7      .3      4
i4      42.244      3.64    7000    195.998     .7      .3      4
i4      46.082      3.642   6800    184.997     .7      .3      4
i4      49.916      3.648   6600    246.942     .7      .3      4
i4      53.758      1.822   6400    220         .7      .3      4
i4      55.676      1.824   6300    195.998     .7      .3      4
i4      57.596      1.824   6200    184.997     .7      .3      4
i4      59.518      1.822   6000    195.998     .7      .3      4
i4      61.444      3.64    6000    195.998     .7      .3      4
i4      65.276      3.648   5800    220         .7      .3      4
i4      69.124      3.64    5600    246.942     .7      .3      4
i4      72.956      3.648   5300    220         .7      .3      4
i4      76.802      1.818   5100    195.998     .7      .3      4
i4      78.716      1.824   5000    246.942     .7      .3      4
i4      80.644      1.816   4900    293.665     .7      .3      4
i4      82.56       1.82    4800    329.628     .7      .3      4
i4      84.48       3.644   4700    293.665     .7      .3      4
i4      88.324      1.816   4500    293.665     .7      .3      4
i4      90.24       1.82    4400    261.626     .7      .3      4
i4      92.158      3.646   4300    195.998     .7      .3      4
i4      96          3.644   4200    246.942     .7      .3      4
i4      99.842      1.818   4400    246.942     .7      .3      4
i4      101.764     1.816   4500    220         .7      .3      4
i4      103.682     1.818   4600    195.998     .7      .3      4
i4      105.6       1.82    4700    184.997     .7      .3      4
i4      107.518     3.646   4800    195.998     .7      .3      4
i4      111.36      3.644   5000    184.997     .7      .3      4
i4      115.198     3.646   5200    195.998     .7      .3      4
i4      119.042     3.642   5300    195.998     .7      .3      4
i4      122.88      1.82    5600    195.998     .7      .3      4
i4      124.802     1.818   5600    195.998     .7      .3      4
i4      126.718     1.822   5800    184.997     .7      .3      4
i4      128.636     1.824   5800    195.998     .7      .3      4
```

```
i4      130.56      3.644   6000    195.998     .7      .3      4
i4      134.402     3.642   6100    195.998     .7      .3      4
i4      138.242     3.642   6300    184.997     .7      .3      4
i4      142.08      3.644   6500    246.942     .7      .3      4
i4      145.916     1.824   6700    220         .7      .3      4
i4      147.842     1.818   6800    195.998     .7      .3      4
i4      149.762     1.818   6900    184.997     .7      .3      4
i4      151.676     1.824   7000    195.998     .7      .3      4
i4      153.602     3.642   7100    195.998     .7      .3      4
i4      157.436     3.648   7300    220         .7      .3      4
i4      161.276     3.648   7500    246.942     .7      .3      4
i4      165.12      3.644   7700    220         .7      .3      4
i4      168.958     1.822   7900    195.998     .7      .3      4
i4      170.88      1.82    7900    246.942     .7      .3      4
i4      172.8       1.82    8100    293.665     .7      .3      4
i4      174.716     1.824   8100    329.628     .7      .3      4
i4      176.638     3.646   8300    293.665     .7      .3      4
i4      180.48      1.82    8400    293.665     .7      .3      4
i4      182.4       1.82    8600    261.626     .7      .3      4
i4      184.316     7.488   8600    195.998     .7      .3      4

;Tuba
;Inst   St          Dur     Amp     Freq        P6      P7      P8
i4      3.84        3.644   9800    195.998     .7      .3      4
i4      7.678       1.822   9700    195.998     .7      .3      4
i4      9.598       1.822   9500    146.832     .7      .3      4
i4      11.522      1.818   9500    164.813     .7      .3      4
i4      13.442      1.818   9300    123.471     .7      .3      4
i4      15.362      3.642   9200    164.813     .7      .3      4
i4      19.204      3.64    9000    146.832     .7      .3      4
i4      23.042      3.642   8800    97.999      .7      .3      4
i4      26.878      3.646   8500    195.998     .7      .3      4
i4      30.716      1.824   8300    195.998     .7      .3      4
i4      32.64       1.82    8200    195.998     .7      .3      4
i4      34.556      1.824   8100    146.832     .7      .3      4
i4      36.482      1.818   7900    164.813     .7      .3      4
i4      38.402      3.642   7900    130.813     .7      .3      4
i4      42.242      3.642   7600    97.999      .7      .3      4
i4      46.082      3.642   7400    146.832     .7      .3      4
i4      49.916      3.648   7200    164.813     .7      .3      4
i4      53.76       1.82    6900    184.997     .7      .3      4
i4      55.678      1.822   6800    195.998     .7      .3      4
i4      57.596      1.824   6700    146.832     .7      .3      4
i4      59.52       1.82    6600    123.471     .7      .3      4
i4      61.442      3.642   6500    130.813     .7      .3      4
i4      65.278      3.646   6300    110         .7      .3      4
i4      69.12       3.644   6000    97.999      .7      .3      4
i4      72.956      3.648   5800    184.997     .7      .3      4
i4      76.8        1.82    5600    195.998     .7      .3      4
i4      78.716      1.824   5400    164.813     .7      .3      4
i4      80.644      1.816   5300    146.832     .7      .3      4
```

i4	82.564	1.816	5200	110	.7	.3	4
i4	84.482	1.818	5100	123.471	.7	.3	4
i4	86.396	1.824	5000	130.813	.7	.3	4
i4	88.316	3.648	4900	146.832	.7	.3	4
i4	92.164	3.64	4700	97.999	.7	.3	4
i4	96.004	3.64	4600	195.998	.7	.3	4
i4	99.838	1.822	4800	195.998	.7	.3	4
i4	101.76	1.82	4900	146.832	.7	.3	4
i4	103.682	1.818	5000	164.813	.7	.3	4
i4	105.598	1.822	5100	123.471	.7	.3	4
i4	107.516	3.648	5200	164.813	.7	.3	4
i4	111.356	3.648	5400	146.832	.7	.3	4
i4	115.204	3.64	5700	97.999	.7	.3	4
i4	119.038	3.646	5800	195.998	.7	.3	4
i4	122.88	1.82	6000	195.998	.7	.3	4
i4	124.804	1.816	6100	195.998	.7	.3	4
i4	126.718	1.822	6300	146.832	.7	.3	4
i4	128.642	1.818	6300	164.813	.7	.3	4
i4	130.562	3.642	6500	130.813	.7	.3	4
i4	134.402	3.642	6600	97.999	.7	.3	4
i4	138.238	3.646	6900	146.832	.7	.3	4
i4	142.08	3.644	7100	164.813	.7	.3	4
i4	145.918	1.822	7300	184.997	.7	.3	4
i4	147.836	1.824	7400	195.998	.7	.3	4
i4	149.76	1.82	7500	146.832	.7	.3	4
i4	151.676	1.824	7600	123.471	.7	.3	4
i4	153.604	3.64	7700	130.813	.7	.3	4
i4	157.436	3.648	7900	110	.7	.3	4
i4	161.284	3.64	8200	97.999	.7	.3	4
i4	165.122	3.642	8300	184.997	.7	.3	4
i4	168.956	1.824	8500	195.998	.7	.3	4
i4	170.876	1.824	8600	164.813	.7	.3	4
i4	172.8	1.82	8800	146.832	.7	.3	4
i4	174.716	1.824	8800	110	.7	.3	4
i4	176.642	1.818	9000	123.471	.7	.3	4
i4	178.564	1.816	9100	130.813	.7	.3	4
i4	180.476	3.648	9100	146.832	.7	.3	4
i4	184.318	7.486	9400	97.999	.7	.3	4

Movement VI, *Used To Be*

Sound Effects

Orchestra:

```
sr              =       44100
kr              =       4410
ksmps           =       10
nchnls          =       2
garvbsig        init    0

        instr   1       ;Space
ifreqscale      =       1
ispecwp         =       0
ktime           line    0, p3, 5
irvbgain        =       p5
ibalance        =       p6
apv             pvoc    ktime, ifreqscale, "universe.pvc", ispecwp
                outs            apv, apv
garvbsig        =       garvbsig*irvbgain
        endin

        instr 2         ;Vocal
a1      diskin          "Used2Bvocal.wav", 2
a2      convle          a1, "pianostr.con"
        outs            a2*.02, a2*.02
        endin

        instr   3       ;Wind
```

SEE EXAMPLE 25, P. 37

```
        instr   4       ;GLOBAL REVERB

irvbtime        =       p4
asig            reverb  garvbsig, irvbtime
                outs    asig, asig
garvbsig        =       0
        endin
```

Footnotes

1. Rumsey, Francis: MIDI Systems & Control, Focal Press; London, England 1990, P. 34

2. Boulanger, Dr. Richard: The Csound Book, MIT Press; Cambridge, MA 2001, preface P. xxxi

3. Rumsey, Francis: MIDI Systems & Control, Focal Press; London, England 1990, P. 35

4. Rowe, Dr. Robert: The Midi Standard, Digitally Controlled Music Systems, NYU; New York 2001

5. Cakewalk Pro Audio 9.0: Screen Graphic, Cakewalk Music Software, Watertown, MA 1999

6. Ibid

7. Ibid

8. Aikin, Jim: MIDI Sequencing For Musicians, The GPI Corporation; Cupertino, CA 1989, P. 38-39

9. Pohlmann, Ken C.: Principles of Digital Audio, McGraw-Hill, Inc.; New York 1995, P. 24

10. Bianchhini, Riccardo & Cipriani, Alessandro: Virtual Sound, Con Tempo; Rome, Italy 2000, P. 1

11. Boulanger, Dr. Richard: The Csound Book, MIT Press; Cambridge, MA 2001, P. 39

12. Bianchhini, Riccardo & Cipriani, Alessandro: Virtual Sound, Con Tempo; Rome, Italy 2000, P. 28

13. Boulanger, Dr. Richard: The Csound Book, MIT Press; Cambridge, MA 2001, P. 659

14. Dolson, Mark: The Phase Vocoder, Computer Music Journal, Vol 10, #4, MIT Press; Cambridge, MA 1986, P. 14

15. Dolson, Mark: The Phase Vocoder, Computer Music Journal, Vol 10, #4, MIT Press; Cambridge, MA 1986, P. 20

16. Dan Ellis and Richard Karpen, Csound Manual version 4.15, PVOC.

17. Boulanger, Dr. Richard: The Csound Book, MIT Press; Cambridge, MA 2001, P. 547

18. Bianchhini, Riccardo & Cipriani, Alessandro: Virtual Sound, Con Tempo; Rome, Italy 2000, P. 159

19. Boulanger, Dr. Richard: The Transformation of Speech Into Music, Ph.D. Thesis, University of California, Music Department; San Diego 1985, P. 74

20. Bianchhini, Riccardo & Cipriani, Alessandro: Virtual Sound, Con Tempo; Rome, Italy 2000, P. 247

21. Boulanger, Dr. Richard: The Csound Book, MIT Press; Cambridge, MA 2001, P. 44

Bibliography

Aikin, Jim: *MIDI Sequencing For Musicians,* The GPI Corporation; Cupertino, CA 1989

Bianchhini, Riccardo & Cipriani, Alessandro: *Virtual Sound,* Con Tempo; Rome, Italy 2000

Boulanger, Dr. Richard: *The Csound Book,* MIT Press; Cambridge, MA 2001

Boulanger, Dr. Richard: *The Transformation of Speech Into Music,* Ph.D. Thesis, University of California, Music Department; San Diego 1985

Cakewalk Pro Audio 9.0: *User's Manual,* Cakewalk Music Software, Watertown, MA 1999

Dodge, C., and Jerse, T.: *Computer Music,* Schirmer Books; New York 1997

Dolson, Mark: *The Phase Vocoder,* Computer Music Journal, Vol 10, #4, MIT Press; Cambridge, MA 1986

Ellis, Dan & Karpen, Richard: *Csound Manual, version 4.15*, 2001

Pohlmann, Ken C.: *Principles of Digital Audio,* McGraw-Hill, Inc.; New York 1995

Rowe, Dr. Robert: *The Midi Standard*, Digitally Controlled Music Systems, NYU; New York 2001

Rumsey, Francis: *MIDI Systems & Control,* Focal Press; London, England 1990