ELECTRONIC AND COMPUTER MUSIC TECHNOLOGY SINCE 1900

Around the time that automobiles were beginning to replace the horse and buggy (c. ~1900), people began to experiment with electrical synthesizers. Even before the vacuum tube was invented, musical instruments were constructed using electrical alternators and telephone receiver/transmitters. Since that time, each arrival of a new technology -- vacuum tubes, solid state devices, computers, VLSI and DSP circuits -- led to scores of inventions designed to provide new opportunities for musicians and composers to create music.

In this chapter we will develop a history of electronic music technology by considering three different approaches to music synthesis: 1) **real-time performance instruments**, 2) **coded-performance machines**, and 3) **sound-processing systems**. These correspond to distinct philosophies with respect to generating music. At the risk of being simplistic, we could assert that real-time performance appeals to practical, "hands-on" musicians, whereas abstract thinkers would prefer coding their scores in some fashion. On the other hand, composer-artists are likely to choose the more leisurely and creative approach encouraged by the possibilities of sound processing.

A **real-time performance instrument** converts a performer's gestures (via his hands, mouth, or feet) directly into sound. Performance interfaces associated with acoustical instruments such as the piano, the guitar, and the clarinet have had their counterparts in electronic synthesizers, although the conventional black-and-white key keyboard is by far the most common mechanism for control. Granted, real-time instruments require a great deal of physical finesse and coordination as well as musical knowledge to perform well, regardless of whether they are acoustic or electronic. However, for many, the immediate feedback obtained from real-time performance together with its superior efficiency in music production are indispensable payoffs with this approach.

Some may wish to circumvent the requirements for performing or improvizing in real-time. One solution is to use a machine which allows the user to define a musical score as a series of instructions to a machine; then the music is played back according to the code. The music box, the player piano, and other various mechanical music contraptions of the 18th and 19th centuries can be considered acoustic precursors of 20th-century electronic coded-performance machines (at least when codes were created in non-real time -- as opposed to "recording in", e.g., the player piano). Coding usually involves making premeditated choices of musical parameters (e.g., pitch and duration) to spell out a musical score as opposed to the spontaneous choices which occur in improvization. Coding could also entail the use of complex algorithms -- perhaps using random numbers or fractals-- to construct musical choices throughout a piece of music.

A concept of music which first became popular in the 20th century (at least in certain circles) is that *all sounds* can be used as musical material. This led to the idea of **music sound processing**, where sounds obtained from a variety of sources are recorded into a machine and distorted, edited, and combined in various ways. It is not necessary for the composer to specify musical

pitch since he/she would work directly with sounds and would have available to him a collection of processing tools used to transform and combine sounds into a final composition. Just as a painter creates an oil painting, a composer can build his composition slowly over time by repeatedly working with his materials. The method does not encourage making a finished score for the composition, as a score is not normally used for its realization. Sound processing was first made possible by the availability of a cheap, editable storage medium, analog magnetic tape, around 1950. A group in Paris, France was the first to exploit this technique; they called it **musique concrète**, and the name has stuck since. Digital computers now provide an alternative: Recently the cost of mass digital storage, analog/digital interfaces, graphic editing, and high speed digital processing has decreased to the point where it is practical for individuals to purchase complete systems.

For a long time, the distinction between these three methods of music generation was very clear and machines fell neatly into just one of the three categories. Now, with computers becoming the universal tool, we are beginning to see more systems which combine two or possibly all three of the methods. For example, there are now score editors which allow the user to enter notes either by "performing in" or by direct (non-real-time) entry using a computer keyboard or mouse.

EARLY REAL-TIME PERFORMANCE INSTRUMENTS (1900 - 1960)

The Telharmonium (circa 1900)

The musical keyboard, with its black and white keys, is probably the most practical and most universal device ever invented for musical performance: It combines tone actuation, pitch selection, and loudness control in a single elegant package. In its simplest electrical implementation each key can be used to select or turn on a individual tuned oscillator. Such a technique was used with the first well-known instrument to produce sounds from electricity, the Telharmonium, invented around 1900 by Thaddeus Cahill in Massachusetts.

Cahill's basic signal generating element was an electrical alternator, which produced a sine wave whose frequency varied according to its speed of rotation. Complex tones were built from the combination of several alternator outputs whose frequencies were arranged in the ratios 1:2:3:4..., i.e., the harmonic overtone series. This method of tone generation is called **additive synthesis**. Around 1860, the famous German scientist Hermann Helmholtz [Helmholtz, 1862, 1954] had shown that all sustained musical tones could be characterized in terms of the strengths of harmonic frequency components. Cahill's concept that music could be elegantly synthesized from collections of sine waves looked very promising. (In fact, this method is still very viable.) Waveform control was to be accomplished by keyboards to select fundamental frequencies (pitches) and draw-bars to control the relative strengths of the harmonics for each pitch.

Alternators of that day were extremely large and costly. In 1906 when Cahill moved his Telharmonium from Holyoke, Mass. to New York City, the total system weighed 200 tons and required 30 box cars to transport it on a train! Fortunately for him, Cahill was very wealthy and (for a time) very successful at raising money for the development of the machine. Even so, the

instrument had many limitations. For example, only harmonics 1 to 6, 8, 10, 12, and 16 were generated for each pitch.

For the time the Telharmonium was a colossal undertaking. Its initial success was evident from the rave reviews (in the New York Times, etc.) it received in response to early demonstrations. For example, Ray Stannard Baker, who heard a Telharmonium concert in Hoyoke in 1906, wrote [Baker, 1906]:

> ...When the music began, it seemed to fill the entire room with singularly clear, sweet, perfect tones. ... It was pure music, conveying musical emotion without interference or diversion. As one listens, the marvel grows upon him ... strangely enough, while it possesses ranges of tones all its own, it can be made to imitate closely other musical instruments: flute, oboe, bugle, French horn, and 'cello best of all ...

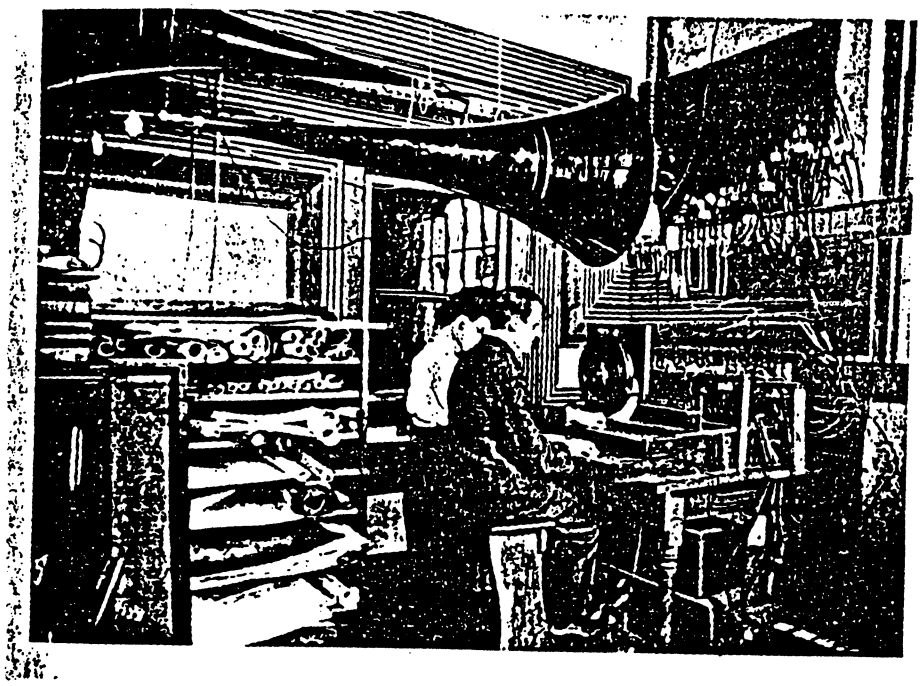
This helped convince financiers to invest in Cahill's plan to distribute music via the telephone system to subscribers (in a fashion similar to today's cablevision). Unfortunately, its popularity soon began to wane. Several sound quality problems had become obvious, including switching transient noises, too few harmonics, insufficient number of voices, etc., which, together with problems of distribution and reports of telephone disruptions, contributed to the failure of this enterprise. Nevertheless, many of the inventions and concepts developed by Cahill inspired electronic musical instrument designs to follow many years later.

Photographs of the Telharmonium control console and some of the "guts" of the machine are shown in Figure 1. The Telharmonium has been written on in detail by Thomas Rhea and Reynold Weidenaar [Rhea, 1972, 1984; Weidenaar, 1988]. Unfortunately, none of the original equipment or recordings of Telharmonium output are known to have survived.

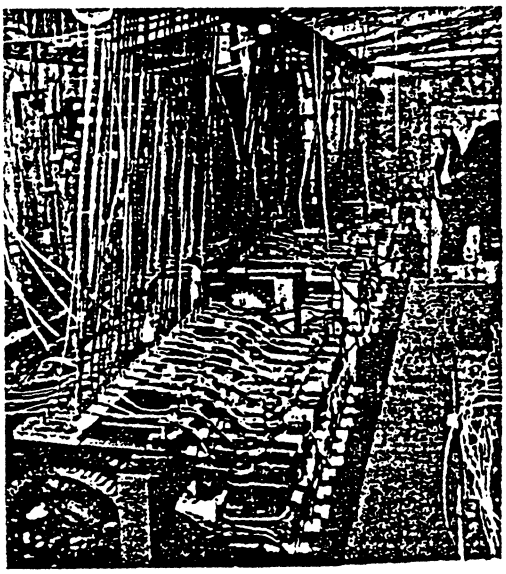
Despite Duddell's invention of a purely electrical musical instrument as early as 1899 and DeForest's invention of the vacuum tube in 1906, no other electrical or electronic instruments of note appeared until the 1920's and none were really commercially successful until the 1930's. Early instruments employed methods ranging from the use of one oscillator per keyboard, with a different capacitor for each key (solo instrument), to the use of several oscillators per key to provide simultaneous notes and variable timbre [Miessner, 1936; LeCaine, 1956].

Three Solo Instruments (1920-1930)

The first electronic instrument of any importance was the Aetherphon or "Theremin" invented by Leon Theremin, a Russian, in the early 1920's. A Theremin consists of two antennae whose effective capacitances are modified by hands in proximity with them. The capacitances serve as components in an high frequency oscillator circuit. One capacitor tunes the frequency of a 170 KHz variable oscillator, and the other controls the amplitude of the oscillator's output. The output is then mixed with that of a fixed 170 KHz oscillator to produce an audible beat tone. Since this is a beat frequency (heterodyne) oscillator, a tremendous range of frequencies is



(a)



(b)

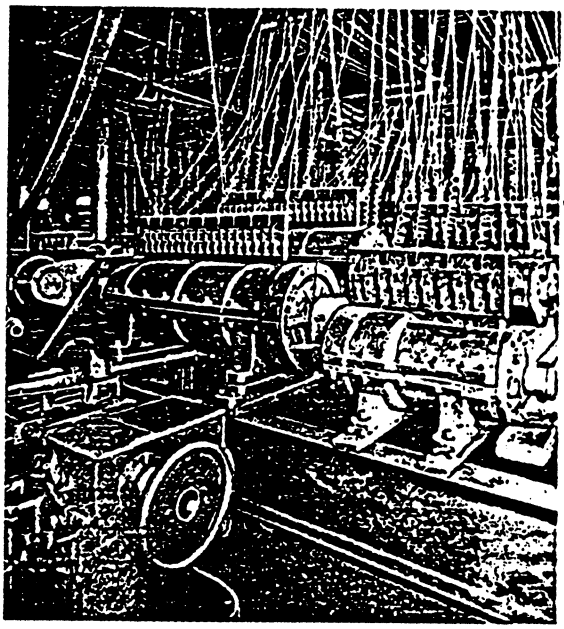


Figure 1. Thaddeus Cahill's Telharmonium. a) Control console, b) Internal generators and wiring.

possible. However, the playing technique is entirely unique and unfamiliar to most musicians. While glissandos are inherently easy to execute, it is very difficult to place the hand so as to control pitch precisely. Most applications of Theremins have been in theatrical situations where eery, siren-like sounds were desired. Between 1924 and 1954 it was used quite often in concert situations and for film scores (e.g., "Spellbound"). Despite its limited, usually bizarre musical use, the Theremin has been a well-known instrument. The gifted artist, Clara Rockmore, made it her principal instrument for concertizing during the '30s, '40s, and '50s, and her work has been preserved on a recording [Rockmore, 1987]. Another dedicated Theremin soloist during this period was Lucy Rosen.

In 1928 the Frenchman Maurice Martenot designed a keyboard-operated solo instrument called the "Ondes Martenot" [Rhea, 1984], which like the Theremin was based on the heterodyne technique. It captured the imagination of several well-known French composers such as Darius Milhaud and Andre Jolivet. From 1947 until his death in the early 1980's, Martenot conducted classes on the performance of this instrument in Paris.

Also in 1928, a German, Friedrich Trautwein, invented a solo instrument, the Trautonium, which was based on a thyratron oscillator [Trautwein, 1930]. Like the Ondes Martenot this instrument attracted much attention amongst serious composers, particularly Paul Hindemith. A later version of the Trautonium, the Mixtur-Trautonium, was developed in the 1950's by Oskar Sala [Sala, 1962] for the synthesis of film music. For sheer timbral quality, Sala's music was not rivaled, at least until the more sophisticated analog and digital synthesizers of the 1970's became available. Three important aspects of the Trautonium were 1) continuous control of pitch, which was enabled by a long resistance wire stretched over a metal plate, and of volume, by means of a pressure-sensitive resistor beneath the plate; 2) generation of a sawtooth waveform containing many harmonics by a thyratron oscillator (The resistive wire varied the rate of voltage buildup across a capacitor. This was applied to the grid of a thyratron tube, which periodically discharged, and thus varied the circuit's output frequency.); and 3) control of tone color by means of several resonance circuits (called "formant" resonators). This method, called **subtractive synthesis**, had become established as a technique for speech vowel generation, but the Trautonium was the first well-known musical instrument to exploit the technique. The control panel of the Mixtur-Trautonium is shown in Figure 2.

The Electronic Organ

Despite advances in electronics, the musical instrument which enjoyed the most commercial success in the pre- and post- World War II eras was based on an electromechanical principal. The Hammond Organ was introduced by Laurens Hammond, an electric clockmaker, in the early 1930's [Dorf, 1968]. The design of this instrument was based on electromagnetic generators, following the general principals of Cahill's Telharmonium, but, of course, in a much more compact form. Serious (classical) composers were not interested in the Hammond, but the instrument was very successfully used for traditional popular music and jazz, and from the 1930's through the 1960's outsold the rest of the organ industry combined.

Like the Telharmonium, the Hammond Organ was an additive synthesis instrument, i.e., separate

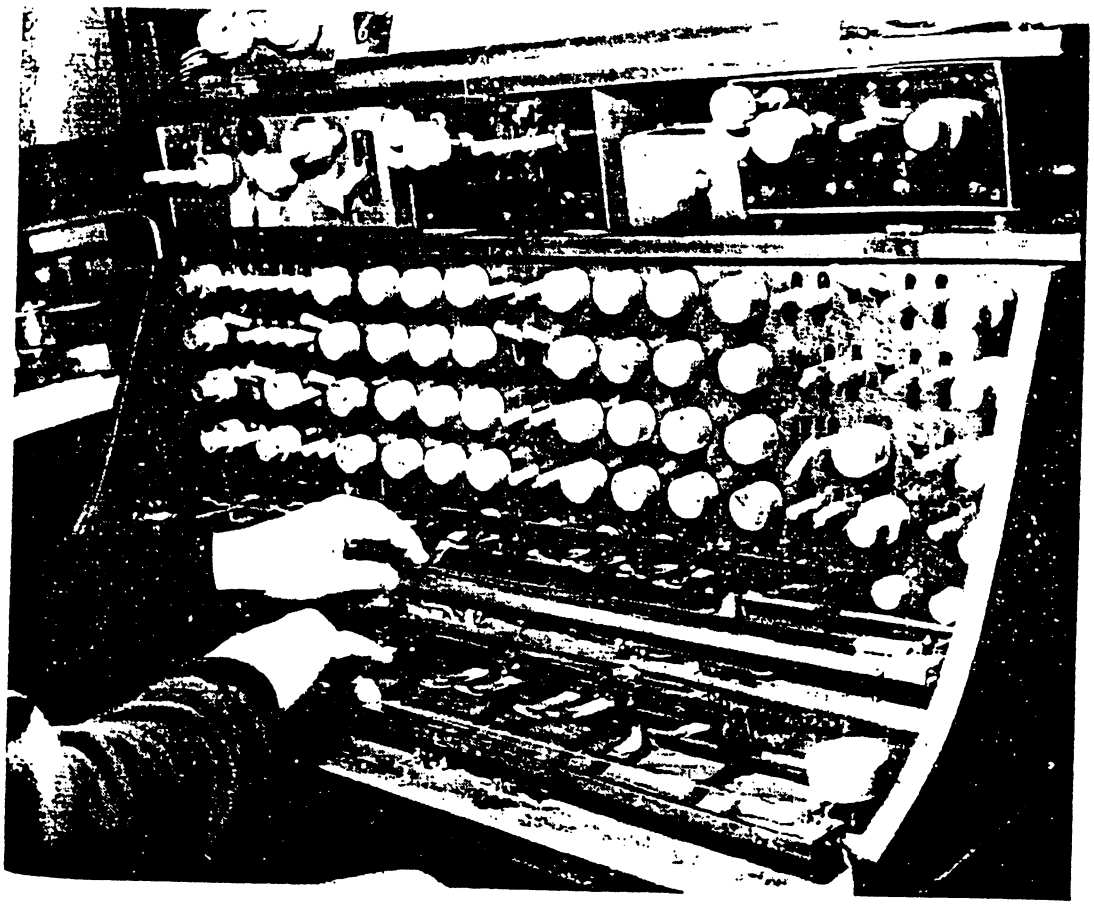
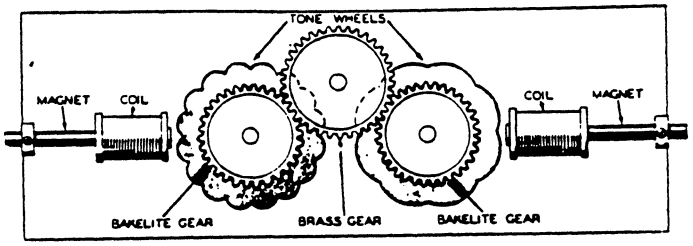


Figure 2. Control console of the Mixtur-Trautonium

(a)



(b)

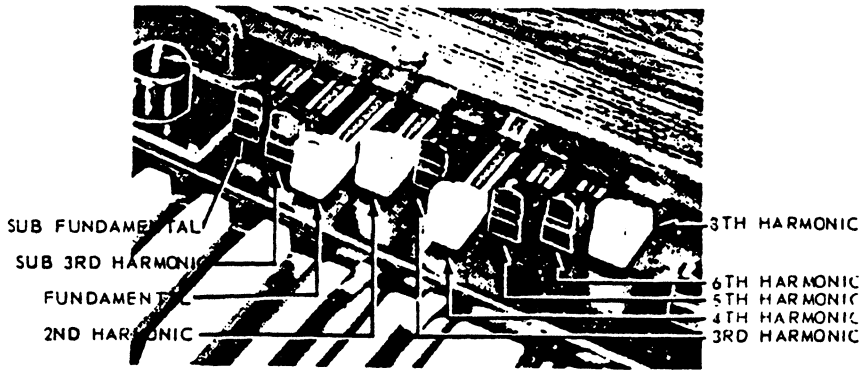


Figure 3. Hammond Organ a) Gear, tone wheel, and coil pickup assembly depicting a 2:1 ratio of frequencies generated. b) Harmonic drawbars mounted just above keyboard.

sine tone components were added together to form each tone. All frequencies generated by the Hammond, whether contained in individual tones or in different tones, were derived from a 60Hz line signal via a synchronous motor and tone wheels as shown in Figure 3a. The number of undulations per rotation cycle of an iron-edged tone wheel determined the frequency of the voltage induced in an associated coil, thus producing a sine wave by means of the variable reluctance principal. There were 91 tone wheels producing 91 different sine wave frequencies in all. Complex tones were formed by combinations of different frequencies, using a series of draw bars. However, since the frequencies were arranged according to approximate equal-tempered tuning (given by the gear ratios used), except for the octaves, the harmonics were out of tune; that is to say, the frequencies were not tuned according to the true harmonic ratios 1:2:3:4:5, etc., but rather the equal-tempered approximations

$$1 : 2 : 2.9966 : 4 : 5.0397, \text{ etc.},$$

(which we will refer to as 1:2:~3:4:~5 below).

The Hammond did not use prime numbered partials above 5 (e.g., "7") because their equal-tempered cousins were judged to be far out of tune, while ~3 and ~5 were thought to be "close enough". While the effect of this difference could become audible to most musically sophisticated listeners, using equal-tempered partials had a distinct advantage: It was practically impossible to produce a dissonant sound on the Hammond bcause there was absolutely no clash between the frequency components of the notes in a chord. "Beats", which would normally occur between colliding overtones of conventional instruments, were virtually non-existent. The result was a very smooth sound (some might even find it dull), all the more so since no frequency above 6000 Hz was generated by the instrument. Generally, "straight tones," where the "out-of-tuneness" of partials would be particularly obvious, were not played on the instrument. Instead, a fair amount of vibrato, tremolo, or "Doppler effect" was employed as a means of tonal embellishment, making it difficult to detect any mistuning of the overtones.

The quality of tones produced by the Hammond could be varied by means of 9 draw bars (shown in Fig. 3b) which varied the relative strengths of the frequencies **f/2, ~3f/2, f, 2f, 3f, 4f, ~5f, 6f,** 8f. Practically speaking, this allowed on the order of 30 to 40 distinctive tone colors. No particular attack/decay envelope was provided, but special percussion circuits were available on some models. These design features were unchanged from the 1930's to the mid 1970's when the Hammond Co. finally discontinued the electromechanical model in favor of a solid state version. Obviously, the original Hammond Organ with its electromechanical tone wheels was a very reliable and popular instrument for many years.

Electronic organs were among the most commonly purchased musical instruments in the 1950's and 1960's, and this popularity caused an American electronic organ industry to flourish. Well-known brands were Baldwin, Wurlitzer, Allen, Gulbransen, Conn, Lowrey, Thomas, and Kimball. The basic technique shared by most of these instruments involved the generation of a "top-octave" equal-tempered scale, which was divided down to form the pitches of the lower octaves. Waveforms rich in harmonics were formed and passed through filters to form various tone colors, which were selected by stops. Thus, unlike the Hammond, these instruments were

for the most part subtractive synthesis machines.

Beginning in the 1960's special effects devices were added to organs, such as percussion rhythm generators and automatic chord accompaniment options, to make them more attractive. In the 1970's features of analog synthesizers were added. This, however, did not stem the tide of interest change by Americans from organs to synthesizers, which occurred during the 1970's and 1980's. The organ had difficulty evolving into a synthesizer since it was based on different aesthetic principles. Its main function was to produce chords rather than a solo voice.

AN EARLY CODED-PERFORMANCE MACHINE: THE RCA SYNTHESIZER (1955)

Musicians and composers are not limited to the possibilities of real-time performance when they use coded-performance techniques. They are free to invent note patterns and other acoustic nuances which they themselves could not perform or may not be performable by any person. Coding the exact pitches, durations, etc. to produce a musical result may be difficult, but this is, after all, what composers have been doing for centuries.

The first electronic coded-performance machine on record was a paper-roll-controlled synthesizer using four oscillators which was demonstrated at the World's Fair in Paris by A. Givelet and A. Coupleux in 1929 [LeCaine, 1956]. This event apparently did not cause any great excitement in the music world.

However, an electronic synthesizer built at RCA [Olson and Belar, 1955] caused a bigger stir. The RCA Synthesizer offered a number of musical possibilities not available with electronic instruments up to that time. According to rumor, the original, rather naive, purpose of the machine was to circumvent a musician's union strike against recording companies by producing popular music direct from the machine without the need for human performers. Indeed, a record featuring simulations of groups playing old-time favorites was released soon after the machine was built. In their 1955 paper Olson and Belar stated that the RCA Synthesizer could "facilitate the production of ... a hit" and the "synthesizer can produce any kind of sound that can be imagined". Later in the article they state that on a statistical basis, based on listening tests, only one of four persons could distinguish synthesizer simulations from identical passages performed on traditional instruments by well known artists!

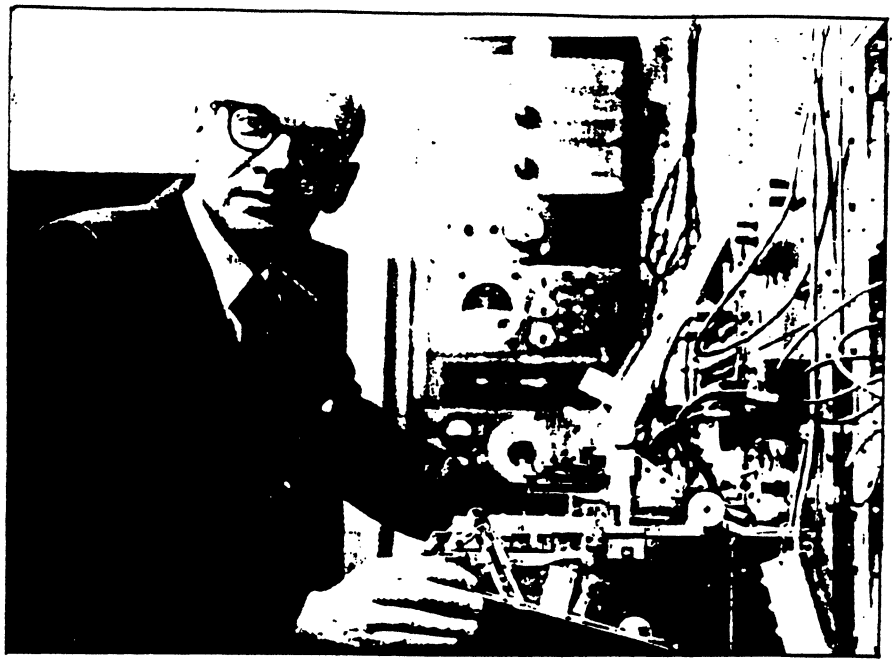
Despite the exaggerated claims about its performance, the RCA Synthesizer was a sophisticated machine for its time. It was also physically very large: It was comprised of vacuum tube and mechanical relay devices mounted in several six-foot high racks. Programming was accomplished by a paper roll with holes punched in it according to a binary code. The original Synthesizer used a disk cutter to record music, two voices at a time. A "multi-track" disk player was used to combine up to six tracks into a single monaural recording.

In 1959 a four voice RCA Synthesizer, the Mark II, was installed at the Columbia-Princeton Electronic Music Center in New York City. The instrument, which later became a gift, was

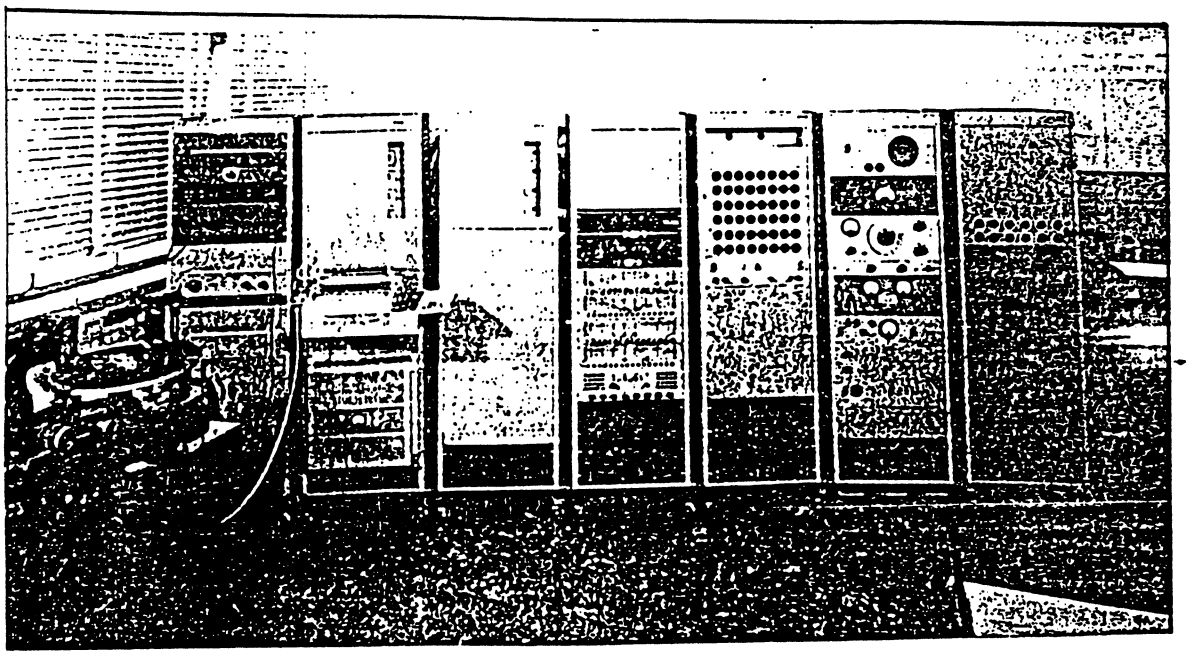
initially loaned to the Center by RCA. By far the most prolific user of the Synthesizer was Milton Babbitt, a professor of music at Princeton University and a composer of "totally serialized" music. To use the machine it was necessary for Babbitt to commute from Princeton to New York on a regular basis. From 1960 to 1968 Babbitt completed at least 4 compositions for the Synthesizer (two of which included soprano voice), and he continued in this vein through the early 1980's. Prof. Babbitt is shown at the console of the RCA Synthesizer in Figure 4a, and Figure 4b shows a view of the entire machine along a wall of the Columbia-Princeton studio.

The Synthesizer allowed for control of frequency, frequency glide, amplitude attack/decay, volume (overall amplitude), and different combinations of filters for tone color control. A block diagram of the Synthesizer is shown in Figure 5. Each of the parameters was coded by means of a 3 or 4 bit code punched in the paper "score". Separate generators, one for each equal-tempered pitch, were selected via relay trees. The generators were very precise tuning fork oscillators; this approach had advantages, but it must be remembered that musical scales other than equal-tempered were impossible. The sine wave output of each basic frequency generator (voice) was applied to eight divider/multipliers which provided selection of one of eight octaves via another relay tree . Thus, the frequency range (of the fundamentals) extended from 23.1 to 5587.7 Hz (F#0 to F8). A special device converted the sine waves to sawtooth waves, which are rich in harmonic overtones, for subsequent subtractive synthesis using the filters.

Note duration was proportional to the number of consecutive paper roll frames coded at the same pitch, and this could be varied by the speed of the travel of the paper. A very nice feature was that the tempo of a composition could be varied independently of the pitches and other parameters over a 4-to-1 ratio from 8 to 32 frames/second.



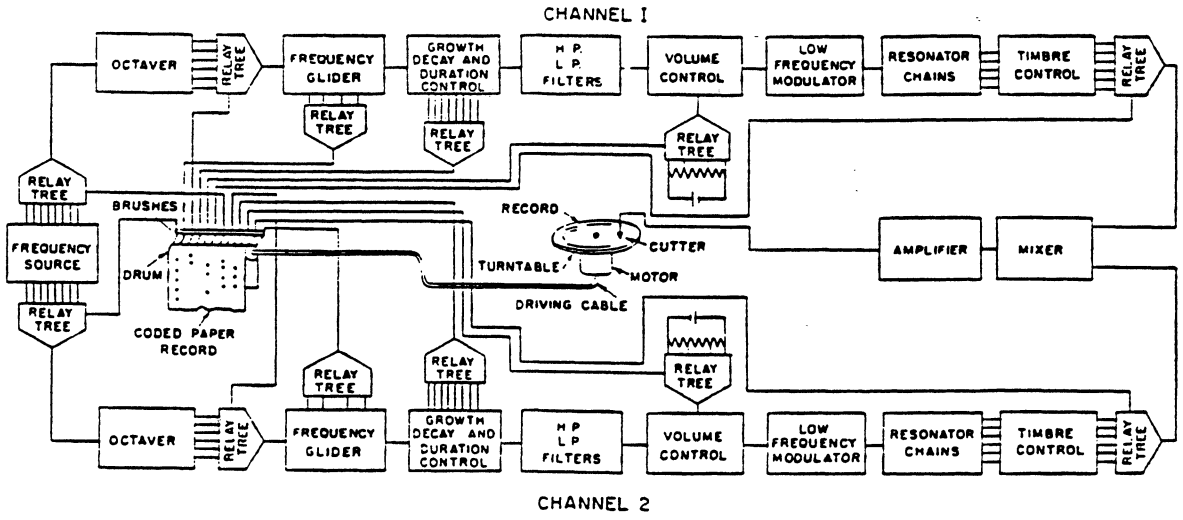
(a)



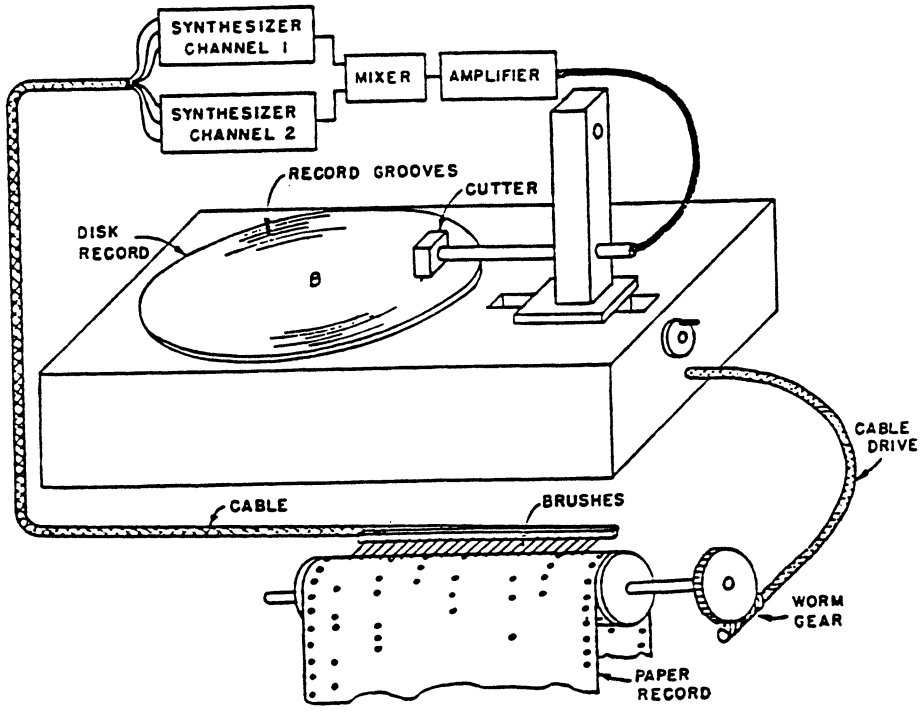
RCA Synthesizer (Full View)

(b)

Figure 4. RCA Synthesizer a) Composer Milton Babbitt, seated before the key-punch board of the RCA Synthesizer at the Columbia-Princeton Music Center. b) RCA Synthesizer (Full View)



(a)



(b)

Figure 5. The RCA Synthesizer a) Block Diagram. b) Punched paper reading device, synthesizer, and disk recording apparatus depicted.

THE CLASSIC STUDIO AND THE SOUND PROCESSING APPROACH (1948 -)

Pierre Schaeffer of Paris, France, was probably the first to use the processing approach in electronic music. He called his music *musique concrète*, first, because of his direct, concrete use of sounds as "objects" in his compositions, and second, because Schaeffer preferred using real or "concrete" sounds as opposed to synthetic ones. While Schaeffer's first experiments (called "studies") used phonograph disk recording techniques (1948), it was really the availability of tape recorders in the early 1950's that stimulated an explosion of activity in electronic music. By the early 1950's Schaeffer's studio in the Office de Radio-Television Francaise was organized around the use of several tape recorders, including some with variable speed control, and various types of filters for modification of sounds. This organization is now known as *Groupe de Recherches Musicales of ORTF*, and it continues to focus on the sound processing approach, although now using digital computer means for processing.

During the 1950's several studios for the production of "tape music" were organized. European studios were mostly housed in radio stations. The earliest ones were at

Paris, France (Groupe de Recherches Musicales of ORTF, since 1951)
Cologne, Germany (Studio fur Elektronische Musik, Westdeutscher Rundfunk, since 1953)
Milan, Italy (Studio di Fonologia, Radio Audizioni Italiane, since 1955)
Warsaw, Poland (Studio Eksperymentaine, Polskie Radio, since 1957)

On the other hand, studios in the U.S. were mainly housed in music departments of various universities:

Columbia University (Tape Music Studio, since 1953) (this became the Columbia-Princeton Electronic Music Center in 1959)
University of Illinois, Urbana, IL (Experimental Music Studios, since 1958)
Brandeis University, Watham, MA (Electronic Music Studio, since 1961)
Yale University, New Haven, CN (Electronic Music Studio, since 1962)

In addition, there were several important independent studios:

Oskar Sala, Berlin (1948-57 private, since 1958 in Hause Mars-Film)
Louis and Bebe Barron, New York City (1948-61)
San Francisco Tape Music Center (1960-68)
Cooperative Studio for Electronic Music, Ann Arbor, Mich. (1958-68)
Independent Electronic Music Center, Trumansburg, N.Y. (1966-72)

While these studios used many different electronic technologies, they all favored the processing approach, centered around the use of analog magnetic tape. The studios themselves were actually rooms housing assemblages of various apparatus based on pre-1960's technology. Sound production was highly dependent on manual knob twisting (for continuous pitch changes, for example), tape splicing (to control the positioning of sound objects in time), and direct sound

processing, most notably by means of electrical filters and reverberators, to achieve sound coloration. The methods might have been tedious and clumsy, but overall they were very effective.

The number of studios grew quite dramatically during the 1960's and early 1970's, but this increase was in large part due to the invention of the voltage-controlled synthesizer, discussed in the next section. A studio using this new equipment might be called a "synthesizer studio", but the synthesizer was generally only used as a special sound generator embedded within the context of a more conventional studio environment [Moog, 1967].

Within 15 years of their birth, tape music studios became referred to as "classic studios", mainly because of the vintage of their technology. Even so, the versatility of the classic tape technique was not supplanted by synthesizers and would not be until computerizd, general-purpose synthesis/editing machines could be created and made available at some reasonable cost. Figure 6 shows two views of a typical electronic tape studio (Institute for Psycho-Acoustics and Electronic Music, Ghent, Belgium, 1973).

Here is a list of equipment which would make up a typical "classic studio":

1) Two or three high quality reel-to-reel tape recorders with splicing blocks. Variable speed and reverse play are very desirable features. Any feature that aids the editing process is very helpful.

2) An audio mixer with several input channels and a lesser number of output channels. This is a device which performs simple weighted addition of several input signals where the weights (gains) applied to each signal are controlled by each user. (Amazingly, practical inexpensive mixers did not appear on the commercial market until the 1970's. Before that, most studios had to build their own.)

3) An audio playback system (amplifiers and speakers).

4) A patch panel for interconnecting units. A switch panel could be substituted, but versatility was usually decreased.

5) One or more microphones for introducing live sounds into the system.

6) A set of electrical filters. For example, Allison Corp. supplied filters which were low pass, band pass, high pass, or band reject with variable cutoff frequencies. These are used for coloring sounds.

7) A "ring modulator", a device which, in effect, performs the instantaneous multiplication of two signals.. When one signal is a sine wave, the result is to shift the spectrum of the other signal according to the frequency of the sine wave.

8) A set of waveform generators. Particularly useful are ones which produce sawtooth and square waves, containing a broad range of harmonics. Sine waves are especially useful if a

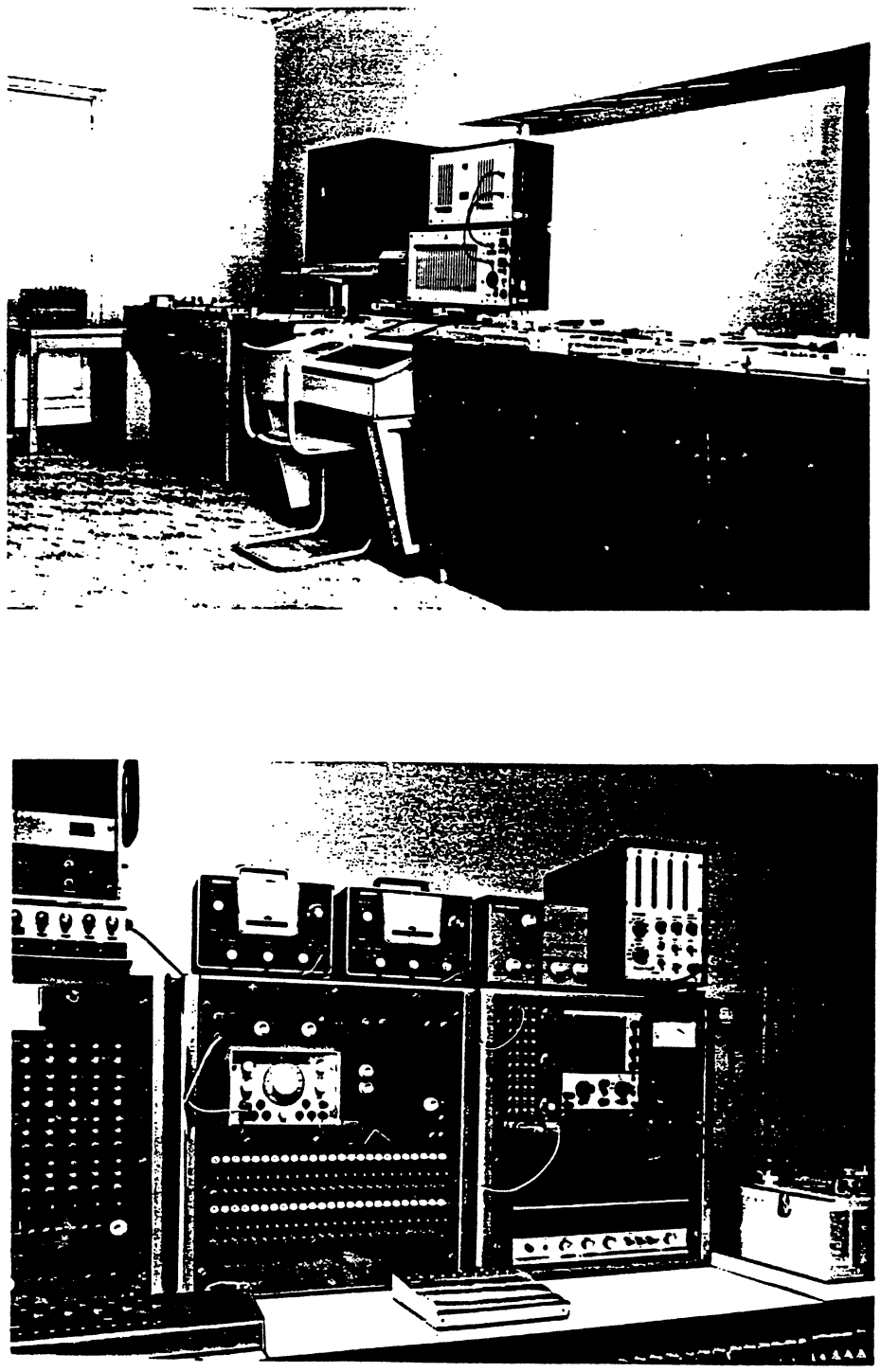


Figure 6. Portions of a "classic" electronic music studio. (Ghent, Belgium, 1973)

number of them are available for creating sound spectra not possible by filtering sawtooth or square waves. Another very useful generator is the white noise generator, whose output signal theoretically contains a continuous density of frequencies across the entire audio band. Unlike the voltage-controlled devices introduced in the mid-1960's, these generators were only capable of manual control.

As previously mentioned, this limited set of equipment -- primitive by today's standards -- had an enormous potential for the production of sounds. But the technique required a great deal of patience, planning (simply "playing around" in the studio seldom led anywhere), and painstaking craftsmanship. Craftsmanship implied precision of execution, and this is one of the most difficult points of the processing technique. An example is the problem of time alignment of the various layers of a musical structure. So long as absolute alignment was not required, there was no problem. (Absolute synchronization was difficult because, for example, it was difficult to lock the speeds of individual tape recorders.) As a compromise, many compositions were created which deliberately allowed for a certain slack in the timing between layers, a situation which would not be tolerated in most traditional music. This is an example where the limitations of the method had a profound influence on the style of the art form.

The reader is referred to articles by Cross [1968] and Luening [1975] for more details on the history of classic electronic music development.

THE SECOND PHASE OF PERFORMANCE INSTRUMENTS: ANALOG SYNTHESIZERS (1964- 198?)

One of the difficulties of most pre-1960 electronic music technology was that musical parameters, such as frequency and amplitude, could be not be automatically programmed. To be sure, remote control was possible, for example, with mechanical relay switches (e.g., the RCA Synthesizer), but the technology was cumbersome and expensive. A technique which had yet to be exploited in music applications was *voltage control*, where a voltage could be used to directly modify a musical parameter. While this method was possible with vacuum tube circuits, it only became practical, in the early 1960's, with the availability of low cost semiconductor circuits . With voltage control it was possible to program the changes of a parameter, such as frequency, using a voltage obtained from another source. Thus,

$$f = G(v)$$

where **f**=frequency, **v**=control voltage, and **G** is a functional relationship between the parameters **f** and **v**.

While in ordinary situations a linear relationship would be appropriate:

$$f = k\,v,$$

For music an exponential relationship is more useful:

$$f = f_0\, 2^{v/v_0},$$

where f_0 = base frequency and v_0=volts-per-octave.

Beginning in 1964 electronic devices designed expressly for music synthesis began to appear on the market. This was spurred by the minaturization and low cost of the new transistors and IC's. Collections of these devices became known as "synthesizers", soon to become "instruments" in their own right. But at first the devices were termed "modules", which implied that they could interconnected in an arbitrary fashion. Unlike the large laboratory generators, supplied by companies like Hewlett-Packard, the new modules were small, voltage-controlled, and music-oriented. Like their lab module predecessors, they featured specific input and output receptacles in order that the output of any device could be connected to the input of any other device. The interconnection of several modules became known as "a patch".

Designs for the first voltage-controlled modules were described by Robert A. Moog at the 1964 Audio Engineering Society Convention in New York (Moog, 1965). {Also described at that meeting was another voltage-controlled instrument called the "Harmonic Tone Generator", designed for additive synthesis [Beauchamp, 1966].} At first, the Moog systems sold rather slowly, mostly to universities who wished to augment their classic studios or start up new ones (university studios began to proliferate in the 1960's) and occasional independent commercial

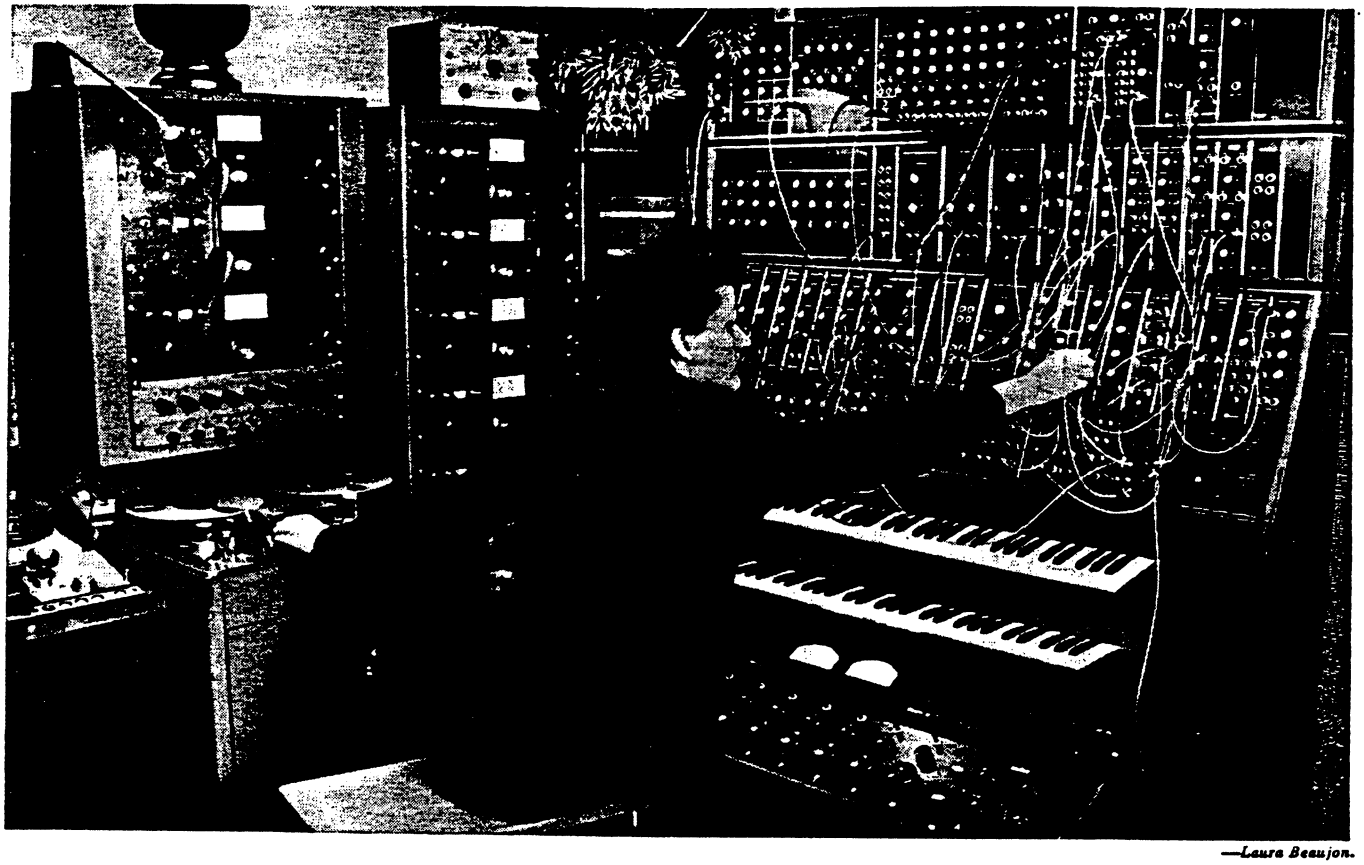
musicians (e.g., Eric Siday of New York, a radio/television composer).

The sales situation changed radically in 1968 when Columbia Records released W. Carlos' "Switched On Bach", a collection of works by J.S. Bach. Carlos was one of the early users of multi-track recorders (see Carlos and Folkman, 1968). Carlos technique was to record these contrapuntal compositions one line at a time (using his 8-track recorder's "sel-sync" feature for time alignment) from a keyboard-driven Moog Synthesizer. Figure 7 shows Carlos working at the synthesizer in the late 1960's. The album featured some very unusual and attractive timbres, which served to delineate the various melodic voices, a property so necessary in much of Bach's music. This record also contained some of the very first examples exploiting Moog's voltage-controlled filter modules, at that time an indispensable tool for generation of brass-like sounds. In fact, the Moog "brass sound" was for several years one of the principal *signatures* of synthesizer sound. The commercial success of this record stimulated a proliferation of recordings using the Moog Synthesizer --mostly in a popular vein -- with such titles as "Moog Espana" and "Music to Moog By". This sudden interest in synthesizers spurred an interest in electronic music in general, and commercial as well as academic activity in this area sharply increased.

A number of other manufacturers such as Buchla, ARP, EMS (London), and ElectroComp entered the electronic music arena during 1966 - 1970, increasing the variety of equipment available, although for the most part these synthesizers offered little conceptual improvement over the Moog. However, Donald Buchla's synthesizer incorporated several interesting new features and had a large impact on electronic music on the West Cost. Buchla's innovations included the first sequencer, a touch-control module, and an improvement in panel layout (i.e., human engineering). The other synthesizers offered improvements in accuracy (particularly the ARP) and decrease in size and price (e.g., the EMS Putney which sold for as little as $400 in 1970 versus $3000 for a typical Moog). Figure 8 illustrates the ARP 2500 and Odyssey synthesizers.

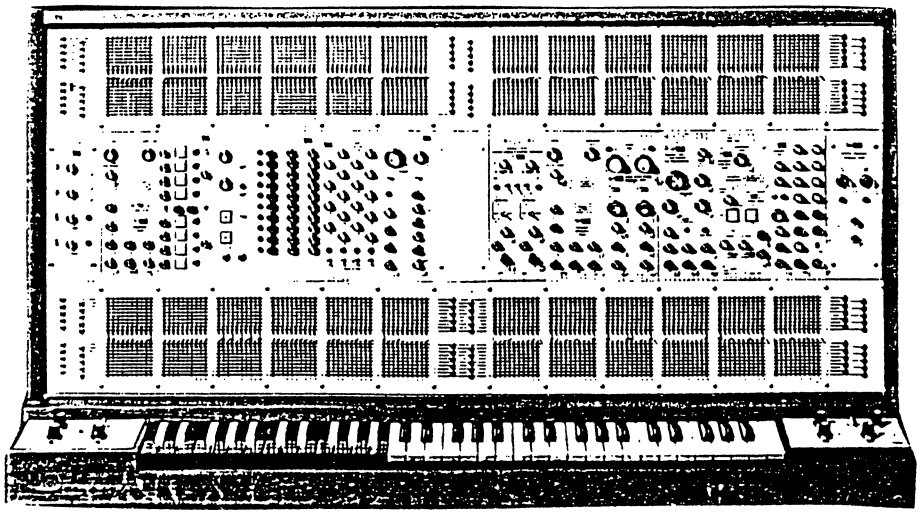
University electronic music studios proliferated in the early 1970's. While these studios were mostly still built around the "classic" ideal, they almost always included one of the large collections of synthesizer modules.

Because synthesizers soon became very popular with pop, rock, and jazz music groups, attention soon became focused on their application for real-time performance applications. Before this time, synthesizers had not really been designed for performance situations, but instead were designed to be very flexible for studio work. The result of this new opportunity for music business profit was a trend toward more portable synthesizers which offered fewer but more quickly obtained possibilities. For example, the ARP Odyssey contained a set of modules (really pseudo-modules, since, unlike the original Moog, they could not be individually removed from the unit) which could be interconnected only by using switches. This confined performance synthesis to a set of "musically useful" patches. In some ways performance synthesizers began to resemble the electronic organ; for example, "stops" were often supplied for selection of specific timbral qualities. This trend was challenged, however, by real-time performances of several composer-performers, such as Morton Subotnick, who exploited the full power of Buchla's

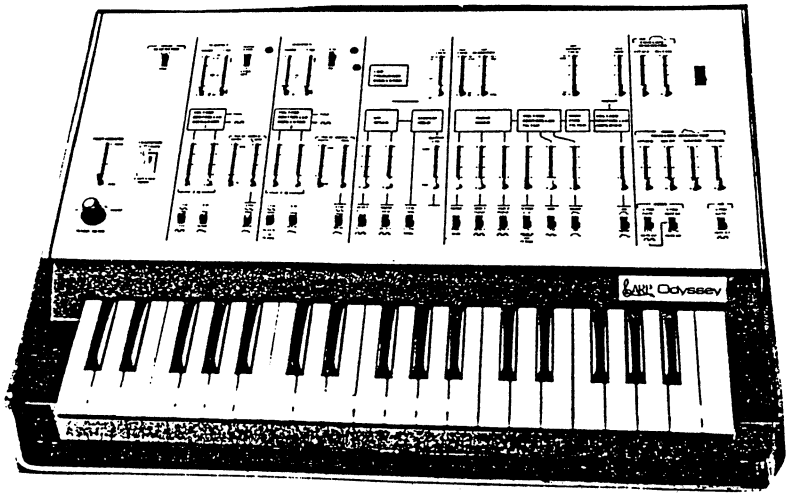


—Laura Beaujon.

Figure 7. Walter Carlos at his Moog Electronic Music Synthesizer. At left is an 8-track recorder which tapes directly from the Synthesizer's output. Above the Synthesizer keyboards are the circuit modules which can be interconnected (as shown here) to generate tones with any desired qualities. Tones are monitored through the loudspeaker barely discernible at the right.



(a)



(b)

Figure 8. ARP Synthesizers : a) ARP 2500 (1970); b) ARP Odyssey (1972)

modular system.

Multi-Voiced Synthesizers

The synthesizer's metamorphosis into a performing instrument required that the number of simultaneous voices it could produce had to be increased from one to several in order that polyphonic and chordal passages could be directly performed. This posed some special problems not encountered by the original single voice synthesizer, such as the necessity for precise tuning among voices and the problem of assigning individual keys for particular voices. Electronic organs, which had a distinct signal source for each key and where each pitch was derived from a common source, did not suffer from this problem. In fact, this solution was adopted for the PolyMoog Synthesizer (marketed in 1976), where a separate LSI synthesizer chip was provided for each key. Polyphonic synthesizers were made possible by increased minaturization of circuitry and use of digital circuits for automatic key-to-voice assignment.

During the late 1970's and early 1980's analog synthesizers moved in the direction of improved compactness and accuracy, increased number of voices, and parameter control by digital circuitry. Because a large number of functional devices could be crammed into a relatively small space, panel design became more refined. For example, many of the same controls (e.g., pitch bend) would normally be applied to all voices. Frequency drift, a problem which plagued the early VCO's, was reduced to a minute level. Voice doubling, the use of two VCO's per voice for warmth, and the use of six or more independent voices became common.

Synthesizer control settings could be stored in a digital memory to be recalled at a later time; this obviated most of the setup time necessary in earlier synthesizers. Furthermore, pitch sequences performed on a keyboard could sometimes be recorded and played back, and synthesizer settings could be modified during performance. Several different timbres could be simultaneously played using a "split keyboard". All of these characteristics enhanced the usefulness of synthesizers as performing instruments. As of 1985 leading manufacturers of analog synthesizers were Oberheim Electronics and Sequential Circuits (USA) and Korg and Roland (Japan).

Despite advances in technology, largely possible due to LSI circuits, relatively little experimentation with the basic design of the analog synthesizer occurred. (An exception was the synthesizer built by a lesser known manufacturer, Serge Systems.) The basic VCO, VCA, VCF, ADSR envelope generator patch became very familiar to a large group of keyboard performers, and their demand encouraged the supply of "more of the same, only better".

CODED-PERFORMANCE USING SOFTWARE SAMPLE GENERATION WITH DIGITAL COMPUTERS (1960 -)

Sometime during 1961, Bell Laboratories at Murray Hill, N.J., issued a record entitled "Music from Mathematics". On one side of the record was a recording of the "Illiac Suite", a piece for string quartet by Lejaren Hiller composed using programs implemented on a computer built at the University of Illinois, the Illiac I. The other items on the record were examples of "computer

music", music generated directly from a computer via a digital-to-analog conversion device at Bell Labs. As such, the record represented two different, but complementary, applications of the computer: music composition and sound synthesis. The latter will be our topic here.

Computer Music Software

About 1960 Max Mathews of Bell Laboratories wrote a program for the IBM 7094 to generate arbitrary waveforms as streams of data [Mathews, 1961, 1963; Roads, 1980]. The power and generality of this program was such that some of its descendents (e.g., Music 4BF and Music 5 (in Fortran) and Music 4C, C Music , and C Sound(in C)) are still in use. These programs take as input **instrument definitions**, a series of statements defining the algorithms used for synthesis, and a **score**, a series of statements which specify the nature of each note to be played. Instrument synthesis algorithms can be defined in terms of interconnected **unit generators**, macros or subroutines designed to output sample data. Mathew's unit generator concept predated the conceptually identical voltage-controlled modules of analog synthesizers by several years. Analog synthesizers and Mathews-style computer synthesis are very similar because in both cases the organization is based on special function building blocks which receive inputs from arbitrary sources and deliver outputs to arbitrary destinations. The computer, like the RCA Synthesizer, also has the power to read coded score input (or even graphic input) and thus makes a very flexible coded-performance machine.

Converting Digital to Audio

The final output of the computer music program is a series of numbers, called "samples", which define the instantaneous audio signal. At least 20,000 samples for each second must be calculated to accurately define a monaural signal, and twice that for stereo. Moreover, each sample must be quantized and coded with at least 12 bits (preferably 16 bits or greater). So the minimum data rate is 0.24 Mbits/second. In a separate step, digital-to-analog conversion, the data must be converted to an analog voltage. This must take place continuously, without break, to avoid clicks in the resulting sound. The actual conversion from binary to analog is accomplished with a relatively inexpensive small component, called a digital-to-analog converter (DAC), which may cost anywhere from $2 (8 bits) to $50 (16 bits). However, much more is needed to make a complete conversion system: a clock oscillator to determine the samples/second rate, sample-hold ("deglitching") amplifiers, multiplexing circuitry for multiple channels, low pass filters to smooth the signals, and most importantly an interface to the computer which supplies the samples. As recently as the mid-80's, the cost of a computer-oriented ADC/DAC system was in the region of several thousand dollars, but recently the use of VLSI digital circuits has reduced the cost considerably.

Most computers cannot compute the data for a reasonably complex sound in real time, so samples must be stored in intermediate bulk memory, such as magnetic tape or disk. Special "interface driver" software is required to transfer data from disk or tape to a DAC interface at audio rates without breaks in the data flow. This is especially difficult to write for multi-tasking computers, but it has been accomplished for several computers. For example, the NeXT computer supports this feature.

Note that for a piece of music of reasonable length --say, 3 minutes -- at least 43.2 Mbits or 5.4 Mbytes of information is needed for intermediate storage. This is easily within the capability of most hard disks and digital tapes, but beyond the storage capacity of most floppy disks. Hal Chamberlin [1981] demonstrated that dual floppy units could be used to achieve sample rates up to 30K samples/sec. While a 1 Mb floppy can store only about 30 seconds of sound, 300 Mbyte disk units are available (costing about $1000) which can store up to 2.5 hours of sound, and a digital magnetic tape (3600 ft., 6250 BPI) can store about the same amount of information.

Once a conversion interface has been established, progress stems from software development and from focusing on the creation of music. Compared to the use of the classic studio and the analog synthesizer, software synthesis requires a more careful academic approach activity--certainly, intuition alone will not suffice. As in the tape studio approach, however, a composition can be built up over a long period of time, parts of it savored, rejected, or modified, before the work takes its final form.

The FM Synthesis Algorithm

A computer synthesis technique which has enjoyed a great deal of popularity is **frequency modulation** (FM) [Chowning, 1973]. In its simplest form, a sine wave is used to modulate the frequency of another sine wave. While a slow rate of modulation would simply produce a vibrato effect, faster rates can produce new and interesting spectra. Actually, by the time this method was introduced, it had already been used for years in voltage-controlled analog synthesizers. However, amongst the users of analog equipment the excitement about FM did not rise to an especially high level for several reasons: Amplitude modulation and frequency modulation were both used in analog synthesizers to produce various effects, but neither method seemed superior to the other. The analog voltage-controlled filter was available for producing many of the sounds that digital FM could produce. Voltage-controlled oscillators used in analog synthesizers had insufficient precision to produce the sounds desired using FM. On the other hand, while digital versions of the analog voltage-controlled filter turned out to be too costly, the FM technique made an ample substitute for cheap production of exotic sounds . It not only could easily produce brass-like and other instrument-like sounds, it could also produce clangorous (bell-like) effects with no increase in complexity. This is one of the few sound synthesis techniques to be covered by a patent, and Yamaha, its present holder, has marketed a line of digital FM musical instruments {e.g., the DX7, introduced in 1983}. Various other synthesizer manufacturers have licensed the patent from Yamaha.

Other Issues

During the 1970's and 1980's a number of centers were established dedicated to the use of computers in music generation and acoustic research. Well-endowed ones were established at Stanford University (CCRMA), University of California at San Diego (CARL), M.I.T., Northwestern University, and IRCAM at Centre Pompidou in Paris, France. Compared to the earlier classic studios these centers were very expensive to set up and maintain. For example, the Computer Audio Research Laboratory (CARL) at U.C.S.D. budgeted $700,000 for the period

1979-81 to set up and maintain their new facility. It took a much lower budget ($40,000) to set up a small computer music studio, the Computer Music Project, at the University of Illinois Urbana-Champaign during 1984-1986.

Each year the International Computer Music Conference is held for the purpose of communication among composers and technologists. A typical conference includes technical paper sessions (including poster sessions), electroacoustic music concerts, demonstrations, and exhibits. Technical sessions may occur on such topics as computer music composition, sound synthesis algorithms, synthesis hardware, computer music systems, acoustics and psychoacoustics, sound analysis, music input languages, graphic representations, digital audio, personal computer applications, computer-assisted instruction, and real-time (live) performance. Since 1993 the location of the conference has been alternating between the Orient, North America, and Europe.

Besides being used to generate music, computers are increasingly employed to analyze, process, and edit acoustically generated sounds. This requires an analog-to-digital converter (ADC), the reverse of a DAC. Most of the problems associated with sustained DAC also hold true for ADC, so that the system design criteria are essentially the same. The use of sound input to a computer, as well as output, opens up a whole new world of possibilities. With the proper input controls, fast enough computation speed, immediacy of playback, and good interactive graphic displays, the computer holds the potential of being the ultimate sound processing machine.

THE THIRD PHASE OF PERFORMANCE INSTRUMENTS: DIGITAL SYNTHESIZERS (1974-)

While analog synthesizers were very successful during the 1960's and 1970's, there were drawbacks limiting their performance. The chief complaint was that they lacked precision, the same complaint which was proffered against analog computers in the 1950's and spurred the development of digital computers. The answer, of course, was to replace the analog modules with digital ones. While that is precisely what Mathews did in his computer music program, his modules were in software, not hardware. If real-time was necessary, a general purpose computer would not suffice; instead, one needed special hardware to compute samples fast enough for real time. With some sacrifice of generality, in the mid-1970's digital synthesizers were designed to perform as sophisticated real-time musical instruments. Commercial products arrived in the early 1980's.

One of the advantages digital synthesizers had over their analog cousins was the ease with which they could generate arbitrary waveforms. With the aid of small microprocessor to perform Fourier synthesis, arbitrary combinations of harmonics could be summed, loaded into wave tables, and played back at any desired frequency. Another distinct advantage was that digital oscillators were inherently free of tuning drift. Finally, noise and distortion were virtually eliminated.

The heart of any digital synthesizer is the digital oscillator or "phase oscillator". Phase is defined

as the relative position in the wave table of the current sample. To create a constant frequency waveform, the phase is incremented by a constant amount at each sample time. However, when the maximum wave table index (i.e., phase) is reached, we "wrap the phase around" to the beginning of the table again. (This concept is easy to understand if we imagine the waveform to be embossed on a cylinder and curving around it, while the phase index is indicated in units around the bottom edge.) Fortunately, wrap-around automatically occurs with a finite binary index register to which we continue to add numbers. When the number is too large for the register, overflow occurs (which is ignored), and a small phase value again occupies the register.

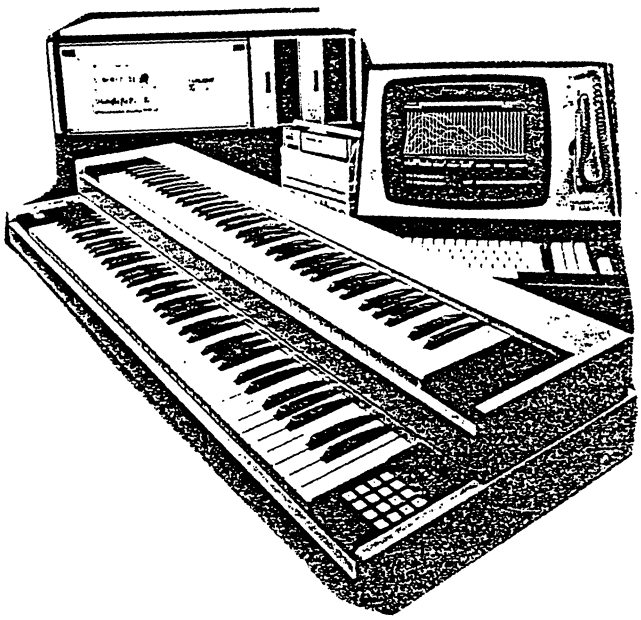
Frequency modulation -- one of the most popular digital synthesizer techniques -- is accomplished by feeding the output of one digital oscillator into the increment input of another. However, the output must be scaled properly, and scaling requires a digital multiplier, for many years a very expensive item. In the 1970's and 80's addition and multiplication were more expensive in digital form than in analog, but if the price were paid, much was gained in precision. For example, it is difficult to control the "initial phase" of an analog oscillator, but this is almost trivial with a digital one. Operations where phase is important become much more accurate.

Because of the expense of digital circuitry and the need for many voices, a digital synthesizer's circuitry must be shared amongst the several voices. This is accomplished by multiplexing techniques, where each voice occupies a "time slot" for accessing a given wave table. Rather than waiting for all register accesses and arithmetic operations to occur during each time slot before the next voice can be served (thus limiting the sample rate), the process is speeded up by "pipeline" techniques, provided there is sufficient hardware duplication.

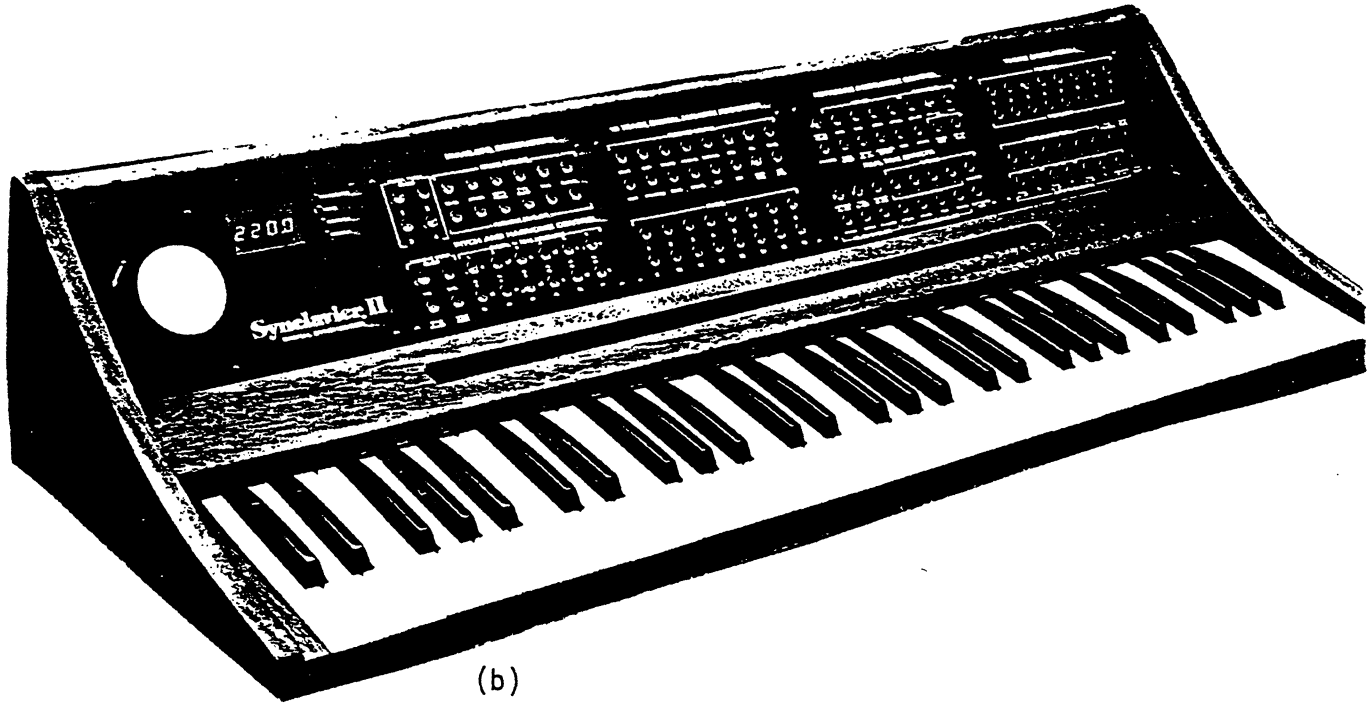
Digital Keyboard Synthesizers

The first successful multi-voice all-digital synthesizer with a keyboard was the Dartmouth Synthesizer, designed by Sidney Alonzo in 1974 [Alonzo et al., 1976]. It was a 16-voice synthesizer controlled by an Alpha-16 minicomputer. The system was capable of splitting its output into four groups of four voices, allocated to one "master station" and 3 remote stations. Software was developed for interactive ear training and music theory exercises. However, all 16 voices could be played back at the master station, where composers could use a music keyboard and a 4-speaker playback system. The two basic synthesis algorithms Alonzo employed were frequency modulation and variable spectrum fixed wave synthesis.

In 1977 the Dartmouth Synthesizer spawned the SynClavier under Alonzo's new firm, New England Digital, Inc. (Norwich, Vermont). The SynClavier was a very versatile computer-controlled keyboard instrument (it used NED's own Able computer for control) and was very low in price ($13,500 up) compared to other digital synthesizers available at that time. It was the first synthesizer to include an internal sequencer with "punch in/punch out" features, emulating the action of multi-track tape recorders, but with much more immediate response. Its keyboard controller displayed an array of lighted buttons for selecting its various modes of performance and included one large spring-loaded analog knob for entry of virtually any parameter. The SynClavier is now considered a high-end instrument intended for advanced studio use. The SynClavier II is shown in Figure 9b.



(a)



(b)

Figure 9. a) The Fairlight Computer Music Instrument (1980); b) The Synclavier II (1980)

Beginning in 1979 several other keyboard digital synthesizers entered the market: The Con Brio (Pasadena), the Fairlight CMI (Australia), and the Crumar GDS (New York). Each of these instruments featured a punch in/punch out sequencer and the ability to modify instrument parameters via a computer terminal or with special controls. They differed in the synthesis methods they had to offer.

The strong professional markets established by New England Digital (SynClavier) -- for FM synthesis -- and Fairlight and E-mu -- for sample synthesis -- created a demand for lower-priced digital keyboard instruments, which firms such as Yamaha, Casio, and Ensoniq were quick to fill. Achieving lower unit cost and miniaturization depended heavily on a firm's ability to design custom LSI chips for music synthesis and to establish high-volume sales to support the high initial investment.

In 1981 Yamaha (Japan) introduced its first FM keyboard instrument, the wood-panelled GS1, selling for $16000. A more portable CE20 Combo Ensemble costing $1400 followed in 1982. Both of these instruments featured selection of a finite set of timbres, although the GS1 set could be varied by insertion of "magnetic voice cards" supplied by Yamaha. Then, in 1983 Yamaha burst onto the amateur/professional market with an array of new products, the most immediately popular of which was the DX7 keyboard synthesizer ($1500). The DX7's features included:

1) Sixteen simultaneous voices of FM (uniform patch).
2) Velocity and *aftertouch* pressure-controlled keyboard.
3) Choice of 32 timbre patches (user or factory programmed).
4) Programmable 5 segment envelopes.
5) Keyboard scaling to allow variation of amplitudes with pitch.
6) MIDI input/output programmability.

The last item on the above list introduced a radically new capability: MIDI (Musical Instrument Digital Interface), a universal standard for externally controlling synthesizers. This standard was originally developed to allow synthesizers from various manufacturers to be performed from a common keyboard or from a stand-alone sequencer. However, it soon became clear that MIDI's role in allowing computers to program musical passages was much more important. We will be discussing the MIDI phenomenon at more length under the section

Computer-Controlled Hybrid Synthesizers.

Yamaha produced several synthesizers closely related to the DX7 during the 1980's, e.g., the DX9 (simplified DX7), DX1 (expanded DX7), CX5M (includes computer for control), TX816 (rack mount expanded version), QX1 and QX7 (stand-alone sequencers), and the DX100 (mini-keyboard version). An upgraded DX7 II was released in 1986.

Casio's products were strictly aimed at the home market for several years, but in 1985 genuinely programmable synthesizers for professional musicians began to appear, such as the CZ-101 (mini-keyboard) and the CZ-1000 (full size). The CZ synthesizers featured **phase distortion** synthesis with 9-segment envelopes. Many effects such as those available on the DX7 could be

produced, but unlike FM, phase distortion can not generate inharmonic partials, an absolute necessity for most percussion sounds.

Sampling Synthesizers

The Fairlight CMI (Computer Music Instrument) (shown in Figure 9a) was the first synthesizer to perform **sampling**, a technique for recording any sound and playing it back at any pitch using its keyboard. It was thus possible to play back a single saxophone tone on every key of the keyboard, even with several simultaneous voices. This also applied to arbitrary sounds such as dog barks! The Fairlight also featured light-pen graphic entry of waveforms and harmonic envelopes, for time-variant additive synthesis. The CMI was a powerful machine, but had a price to match ($28,000). However, lesser expensive sampling synthesizers were soon to be offered by other firms such as E-mu Systems (the Emulator, $8000, 1981), 360 Systems (Digital Keyboard, $3500, 1983), Ensoniq (Mirage, $1700, 1985).

Much ballyhoo accompanied the announcement of the Kurzweill 250 synthesizer in 1983. Although advertisements claimed that their proprietary Contoured Sound Modelling method was based on "insights gained from the field of Artificial Intelligence", it was clear that this was, in fact, another sampling keyboard synthesizer which utilized techniques similar to those pioneered by Fairlight and E-mu. This is not to degrade the Kurzweill as a product, as it was reputed to be a very well engineered instrument, featuring a weighted piano-like keyboard with velocity control, large library of sampled sounds (the acoustic piano sound was very convincing), and an internal sequencer capable of programming 12 different timbres and up to 12,000 notes. The Kurzweill 250 also was a rather expensive instrument (approx. $14,000) and was primarily used for performance and recording studio work.

In 1985, Ensoniq (Malvern, Penn.) crashed the sampling instrument price barrier when it announced its Mirage model for $1700. The Mirage contained memory for 144,000 8-bit (floating-point) samples, which could be split into as many as 16 different wave tables obtained from recorded sounds. When a key was held down, there was a problem with extending the duration of playback beyond what was allowed by the internal memory. This was circumvented by a technique known as "looping", whereby a portion of the wave table was recycled continuously. (Looping was actually pioneered by Fairlight in 1979, with little fanfare.) After key release, decay was handled by two envelope generators controlling an amplifier/filter combination. While the Mirage might be thought of as having a hybrid digital/analog circuit, the inclusion of analog components is really just an expedient solution to the problem of amplitude and anti-aliasing control.

Recent Digital Synthesizers

In the late 1980's and early 1990's several Japanese firms such as Yamaha, Roland, Korg, and Kawai began releasing new digital synthesizers at a dizzying rate. There was a general trend to include more and more features within a keyboard or rack-mount synthesizer. For example, sequencers and MIDI-programmable reverberation effects were included in some synthesizers. The direction was to create a synthesis "workstation" which could embody all aspects of

synthesis. In 1990 Yamaha announced that it was discontinuing most of the DX/TX series of FM synthesizers and was replacing them by a new SY/TG series. These synthesizers combined the frequency modulation technique and "Advanced Wave Memory", a form of sampling technique; vector control was used to combine these techniques, in an effort to provide a large palette of sound possibilities. Keyboard performance emulation of some acoustic voices reached a new level of quality. A further step in realism was realized in 1993 with Yamaha's "virtual acoustics" synthesizer, the VL1, based on work on **physical modelling** at Stanford University [e. g., Smith, 1992].

Micro-Programmable Digital Synthesizers

This class of synthesizers is distinct from that of the previous sections in that they are capable of synthesis algorithms not envisioned by their designers. They almost always are controlled by a computer, but keyboards or other real-time input devices may be used as control devices.. In addition, they may be capable of processing external sounds. A particular synthesis or processing algorithm must be down-loaded to the machine in order for it to "come to life". In essence, such a synthesizer is a special-purpose, very fast computer, capable of generating or processing musical sound in real time. While the first examples of such synthesizers were built using gate level components, later models used single chip DSPs (digital signal processors).

The design for a powerful "digital signal synthesizer" was announced in 1974 by Peter Samson of Systems Concepts (San Francisco). This was intended to be controlled by a computer for professional music applications. As an example of its power, up to 256 simultaneous voices could be generated in real time. The design's objective, arrived at in cooperation with researchers at Stanford University, was to replace the function of software waveform synthesis employed on their heretofore-used general-purpose computer (a DEC PDP6) with attached DAC. The only model ever built was delivered to Stanford's Center for Computer Research in Music and Acoustics (CCRMA) in 1977, but it served as CCRMA's principal sound source until 1990. Indeed, judging from CCRMA's music productivity during this time period, this synthesizer proved to be one of the most powerful tools for computer music composition ever created.

In 1977 H.G. Alles of Bell Laboratories and P. di Giugno of IRCAM in Paris announced their collaborative development of a digital synthesizer card which featured 64 FM oscillators (32 K samples/sec each), 128 ramp generators, and 15 accumulator registers for oscillator interconnection [Alles & di Giugno, 1977]. The card was designed to interface to an LSI-11 microcomputer. Alles built a "portable digital synthesis system" containing an LSI-11, two floppy disks, ASCII-graphics video terminal, ASCII keyboard, 2-manual music keyboard, 72 slide controls, four 3-axis joy sticks, and a very comprehensive 1400 IC digital sound synthesizer/processor with both DAC output and ADC input all in one package weighing about 300 lbs. (See Figure 10.) At least two models were built, but it seems that the full potential of this machine was not reached due to lack of adequate software to drive it. In the meantime, di Giugno developed a series of digital synthesizers which were dubbed the "4B Machine" (1977), the "4C Machine" (1979), and the "4X Machine" (1981). Again, these were very powerful synthesizers, and, although initially there were some problems in developing good software for

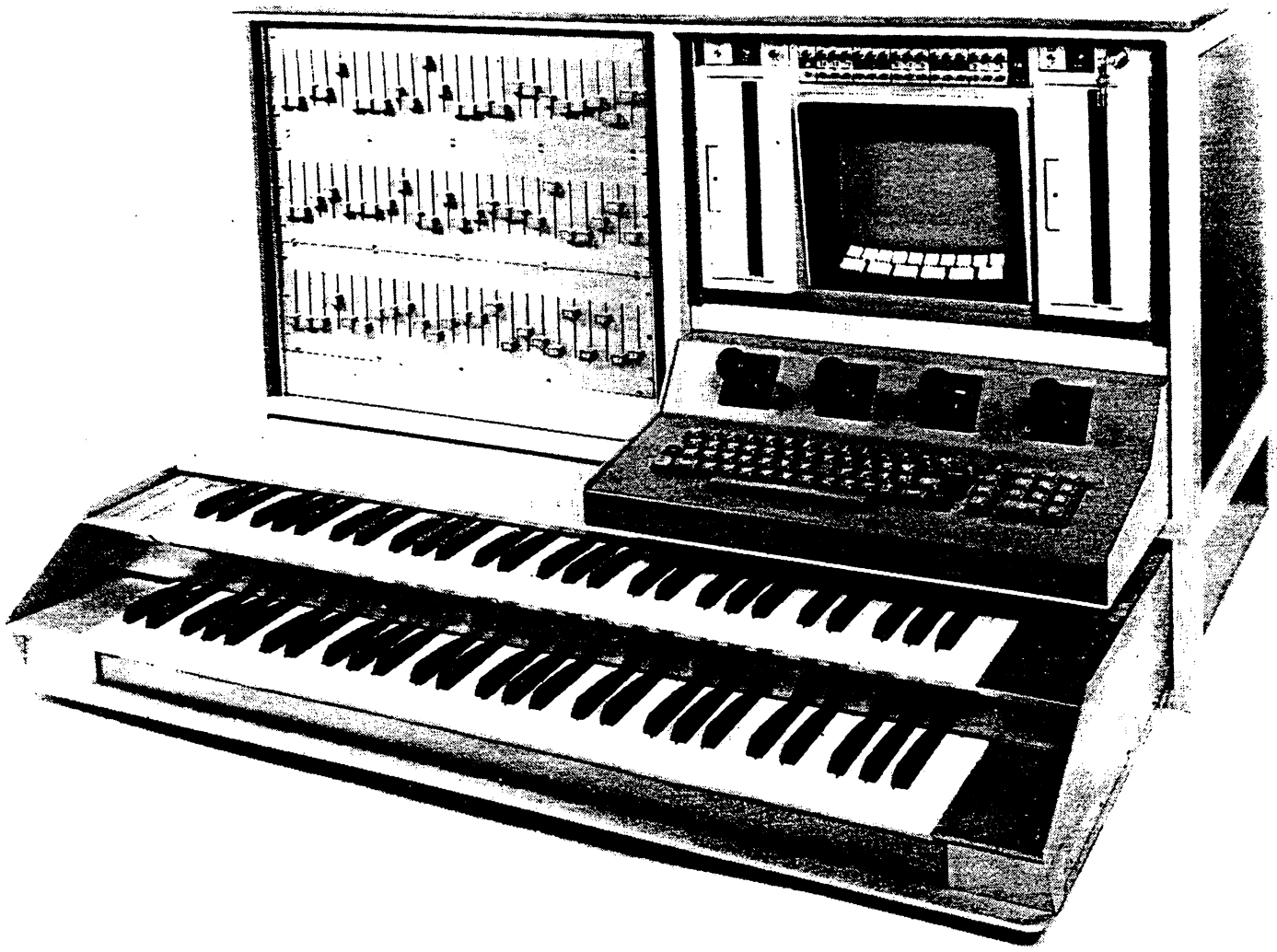


Figure 10. The H.G. Alles Synthesizer (AT&T Bell Labs, 1977)

them, eventually some very reasonable solutions were evolved. In the early 1980's a number of well known composers, including Morton Subotnick, Roger Reynolds, and Pierre Boulez, worked with these machines and produced compositions at IRCAM. Boulez's *Repons*, a composition which employs the 4X synthesizer was premiered at the 1984 ICMC in Paris.

An exceptional digital synthesizer, the DMX-1000 [Walraff, 1979], was introduced in 1979 by Digital Music Systems, Inc. (Boston, Mass.). This was a small special-purpose computer which executed microcode (not including branches) at a fixed speed of 200 nanosecond per instruction. The sample rate was determined by the number of instructions used in a synthesis loop. Using 256 instructions per sample, a sample rate of 19.2 KHz was achieved while simulating, for example, 16 envelope-controlled oscillators in real time. A built-in 16-bit DAC was used to produce the actual sound. The machine was not easy to program, but a language, *Music-1000*, similar to the Music 4 genre of languages (see the previous section *Computer Music Software*), was written for a PDP-11 computer, which in turn controlled the DMX. A complete system (called the DMX-1010), including a PDP-11/03 computer, dual floppy disk, display terminal, control panel, the DMX-1000, and software was offered for $35,300 in 1983. An advantage of this type of machine over other digital synthesizers was that, within its speed limitations, it was capable of arbitrary synthesis micro-programs, enabling the user to experiment with totally new methods of synthesis. Several of these systems were sold worldwide. Barry Truax was a notable user who composed on it using a special method of synthesis called *granular synthesis* [Truax, 1988]. The DMX-1000 could be considered the precursor of what was later called the **DSP**.

At the University of Illinois (Urbana-Champaign) in 1984 Lippold Haken and Kurt Hebel demonstrated their *Platypus* microprogrammable synthesizer [Haken and Hebel, 1987]. The *Platypus* was controlled by an Apple Macintosh computer and executed instructions at a rate of 20 MIPS. One important improvement over the DMX-1000 was the Platypus ability to operate in either real or slower-than-real time. Thus, it could store intermediate results in memory and process *sound objects* in a fashion analogous to a studio for musique concrète. The *Kyma* graphic interface language, written by Carla Scaletti [Scaletti, 1989], allows the user to patch together pre-coded generators and processes in an a arbitrary fashion. (Roughly 30 to 50 fairly complex instruments can be generated in real time.) Thus, a large variety of instruments can be designed by the user. *Kyma* runs under the object-oriented computer language Smalltalk on Macintosh computers and in theory could be run on any computer which offers the Smalltalk environment. In 1990 Scaletti and Hebel formed a company, Symbolic Sound Corporation, to market *Kyma* and a new synthesizer replacing the *Platypus*, the *Capybara*, based on the Motorola 56001 DSP chip. By 1995 the Kyma System had developed to the point where it included spectral analysis and synthesis as well as related functions such as spectral morphing in its repertoire of built-in functions [Miller, 1995]. Thus, the Kyma System has entered the musical sound processing arena.

In the meantime, during the 1980s microprogrammable DSP (digital signal processor) chips were manufactured in large quantities and were incorporated in a wide variety of products, including audio products. For example, in reverberator applications different microprograms could be used to define the quality of reverberation (e.g., "small room" vs. "large room"); the algorithm for reverberating a particular sound would be put in place when the microcode was downloaded to

the DSP. DSP chips powerful enough for general-purpose music processing become available in the late 1980's.

While most early DSPs used fixed point arithmetic, several floating-point chips became available in the late 1980s (e.g., the AT&T DSP32 rated at 12 MFLOPS). Floating-point is very advantageous for computer music applications, as it frees the programmer from concern with most scaling problems, and thus speeds up code development; also the code takes up considerably less memory. In 1989 two manufacturers released products incorporating the AT&T DSP32 on a single board together with memory and high quality sound I/O, Ariel Corp. (for the IBM PC family) and Spectral Innovations (for the Macintosh II). The Spectral Innovations product included a full line of software to perform various signal processing functions (including graphics).

In 1989 the Unix-based NeXT Computer (with Steve Jobs as CEO) was designed to include D/A converters and a Motorola 56001 DSP together with "Sound Kit" and "Music Kit" software for development of new sound synthesis and processing applications [Jaffe, 1989]. During 1989 - 1993 a plethora of sonic-based applications were written for the NeXT. Unfortunately, NeXT Inc. stopped manufacturing their special computer hardware in 1993. While the NeXTStep operating system was ported to other computers including PC-compatibles (Intel), Sun Sparc, and HP, the integrated environment including computer, hard drive, D/As, and DSP was difficult to duplicate on other platforms.

COMPUTER-CONTROLLED HYBRID SYNTHESIZERS (1968 - 198?)

Early Hybrids

"Hybrid synthesizer" originally referred to a music machine consisting of a digital computer attached to an analog synthesizer, although the idea can be extended to include digital synthesizers as well. The use of a computer to control an analog system is fraught with many difficulties and, therefore, has had, historically speaking, a low impact. Nevertheless, literally hundreds of computer/analog hybrid systems were constructed during a period extending from the late 1970s through the 1980s. Today, because of the prevalence of MIDI, the use of computers to control digital synthesizers has become commonplace.

One of the first hybrids was built at Electronic Music Studios (EMS), a government-sponsored center in Stockholm, using a PDP-15 computer to manipulate an analog synthesizer of special design [Wiggen, 1972]. Although the synthesizer utilized solid state rather than vacuum tube technology, its general design resembled that of the RCA Synthesizer. Switches rather than control voltages were used to program the synthesizer, which unfortunately did not take advantage of contemporary voltage-control technology. There was an advantage with this scheme, however, in that, unlike most analog VCOs, the oscillators were very precise and did not suffer from drift. In addition, the user interface consisted of a large group of touch-operated controls, providing for immediate human input. In performance mode, the PDP-15 computer was called upon to generate an immense amount of control data, rivalling that of the data needed to produce actual sound vibrations.

About the same time, a hybrid system called GROOVE was developed at Bell Laboratories in New Jersy [Mathews and Moore, 1970]. A Honeywell DDP-224 computer was used to control a bank of voltage-controlled devices, some of which were Moog synthesizer modules. Most interesting were the methods used to interact with the machine. Control functions were stored on disk as continuous graphs, and convenient methods were devised for generating and editing these functions. A music keyboard was interfaced with the computer so that the system could be played in real time. Also, a three-dimensional control stick was incorporated which allowed for simultaneous control of tempo, loudness, and one other parameter chosen by the user, which could be varied while a precoded piece was being performed. The person controlling the stick became, in effect, the "conductor" of the composition. Thus, the system was both a real-time performance and coded-performance system. Because of the many interactive techniques developed, the GROOVE system was the forerunner of other less costly digital systems introduced later.

Another early hybrid system of note was due to Peter Zinovieff in London (Putney), England. A man of independent means, Zinovieff built a complete computer music studio in the lower level of his home in Putney on the Thames River [Zinovieff, 1969]. The controlling computers were the modest: two DEC PDP8s with a single 32 K hard disk. To these were added a great deal of analog equipment designed by Zinovieff's assistant, the prolific designer, David Cockerell. The approach to synthesis was very eclectic, favoring the production of a wide variety of sounds. The studio was used by several British composers during the 1970's. Zinovieff and Cockerell also formed a company, EMS London, which marketed a line of analog synthesizers (e.g., the Putney and Synthi) with the idea of feeding profits back into further development of the studio.

Despite the attractiveness of combining computers with analog synthesizers, for a number of reasons they made for a rather poor marriage. First, when one uses a computer one comes to expect absolute accuracy and repeatability, which are not fortes of analog circuits (although the later temperature-compensatedcircuits were much improved over early versions where pitch was notoriously unstable and constant retuning was necessary). Second, it was difficult to control minute details in an analog system, such as the details of an attack transient, in order that they could be programmed by a computer. Third, to build up complexity in an analog system, one was essentially forced to duplicate hardware, as compared to a digital system where it is more natural to time-share elementary structures. The latter property of digital systems has led to their superior compactness and ease of manufacture.

Three other hybrid systems developed at universities which were interesting for their use of graphics and computer control should be noted: The SSSP (Structured Sound Synthesis Project) synthesizer at the University of Toronto [Buxton, 1978], used a variety of graphic techniques to evolve scores for driving their digital synthesizer. The PLACOMP system at the University of Illinois used initially a Texas Instruments TI980A computer [Beauchamp, Pohlmann, and Chapman, 1976] and later the PLATO graphic time-sharing system for control of a hybrid synthesizer [Murray, Beauchamp, and Loitz, 1978]. The IMS PLATO music system, also at UIUC, was a 16-voice system primarily used for computer-assisted music instruction and musicology applications [Schmid and Haken, 1984; Scaletti, 1985].

Computer-Controlled Synthesizers Using MIDI

The Musical Instrument Digital Interface is a digital data communication standard developed by a consortium of manufacturers in 1983 for the purpose of allowing one firm's keyboard or sequencer to control another's synthesizer or to connect several synthesizers together [International MIDI Association, 1984]. However, it immediately became obvious that computers could control synthesizers which were equipped with MIDI interfaces. This has opened up computer music to amateur musicians and computer enthusiasts who can afford moderately-priced commercial equipment and created a new industry based on software for MIDI applications.

Prior to MIDI there were a few companies who supplied low cost computer music software and hardware, primarily for the Apple II and Commodore 64 computers. The Apple II could be outfitted with special music circuitry, e.g., the Mountain Music Synthesizer board, and companies such alphaSyntauri and Soundchaser supplied integrated software/hardware packages with music keyboards, based on this combination. However, with the proliferation of MIDI synthesizers available, the software industry was stimulated considerably, and a large number of new, small firms have suddenly sprung into operation. The advent of MIDI transformed a vertical market, where a few large companies supplied products which only worked with that company's software, to a horizontal market, where any musically-inclined programmer could write software to work with a large number of synthesizers. The chief stimulating factors were the low incremental cost of the hardware and software needed to produce music and the availability of products from a large number of different sources.

During 1984-85 commercial software became available (principally for the Apple Macintosh and IBM PC computers, but also for the Atari ST and Amiga computers) to 1) emulate a general-purpose sequencer (i.e., MIDI input from a keyboard and MIDI output to a synthesizer), 2) edit synthesizer parameters, 3) display and print music notation, and 4) convert between notation and MIDI note data. One of the first MIDI-compatible programs to arrive, which combined all of these features, was Standard Productions' *Personal Composer* program (written by Jim Miller) for the IBM PC. Suitably equipped with a PC, a Roland MPU-401 interface (or equivalent), and a MIDI synthesizer, one could enter a score via the keyboard, edit the score (using a menu selection technique), and then immediately play the edited version back on a synthesizer. In addition, the program provided means for designing patches on a DX7 synthesizer and printing scores. Most subsequent programs have been more specialized. By 1989 programs capable of publication quality notation had been developed; two of note are *Finale* (Coda Music Software) for the Apple Macintosh computer and *Score* (Passport Designs) for the IBM PC compatible family of machines running MS-DOS. Sequencer programs also matured; two of the most sophisticated are *Vision* and *MAX* (both marketed by Op Code Systems).

Inexpensive memory-abundant sampling synthesizers appeared in 1985, opening up another area for MIDI related software. Via a MIDI coding mode called "system exclusive" it became possible to transfer sample data between a computer and a synthesizer. Thus, a sampling

synthesizer could be used as a special purpose ADC or DAC with the computer. Signal editing is provided by programs such as *Sound Designer* by DigiDesign for the Macintosh and Atari ST computers. A sound signal is first recorded into a sampling synthesizer and then transferred to the computer via the MIDI path. Using graphics and mouse interaction, the computer can process this data to improve its usefulness for synthesis, such as by defining appropriate waveform looping points and envelopes to define decay patterns. The computer can also synthesize new sounds to be downloaded to the sampler and then played under MIDI control.

Several synthesizers became available in rack mount versions in the mid-1980s. For example, the Yamaha TX81Z was an 8 voice, multi-timbre FM synthesizer that sold for around $400. Note that the combination of a computer, a rack mount synthesizer, and some sequencer software constituted a full-fledged coded-performance synthesis machine. On the other hand, a keyboard controller which outputs MIDI data could be directed either toward the synthesizer or to the computer to input parts one line at a time. The beauty of this approach is its modularity. In 1988 a MIDI file standard was introduced [International MIDI Association, 1988]. This meant that one could create a MIDI file using a sequencer from one firm and play it back or edit it using another firm's software.

One current problem with MIDI systems is that the software frequently doesn't know about the hardware. For example, if the patch parameter settings of a particular synthesizer are inadvertently changed, the result of playing a MIDI file may be completely wrong. This can be circumvented by use of programs which save the synthesizer's internal state in a file and can restore this information later on -- programs called patch/librarians. Even so, there is problem when a computer file representing a composer's composition does not contain enough data to replicate the composition but must depend on a particular synthesizer being attached to the computer with its patch parameters set in a particular way.

Another problem with MIDI is that it is difficult to change synthesizer parameters other than pitch and amplitude from one note to the next, unless a different "channel" is selected. The ability to program any parameter of any voice with high precicion is taken for granted by practitioners of software synthesis. Unfortunately, most MIDI timbral parameters (e.g., attack rate) are handled through a special code called "system exclusive", which is so data intensive that it may interfere with the normal MIDI data stream, and, as a result, notes could possibly be lost. "Microtones " (pitches in between the normal equal-tempered notes) can be handled through either the *pitch bend* parameter or via special synthesizer-specific voice tuning parameters, but there is currently no industry standard for handling microtones and usually pitch-bend is applied equally to all voices, which defeats the purpose of microtones.

SUMMARY

The development of electrical/electronic music began in 1900 with Cahill's promising but ill-fated venture. This was followed by a period of slow development dominated by performance instruments, chiefly the electronic organ. However, in the 1950's the availability of magnetic tape encouraged composers to exploit the possibilities of electronic sound synthesis and processing. The classic tape studio, the analog synthesizer, the main-frame computer and DAC,

and the digital microcomputer/synthesizer combination have each, in turn, had an impact. Since the 1960s a vast number of synthesizers have been built and put to use. This activity has been primarily due to the advent of solid state devices and the extreme minaturization of sophisticated devices. Whereas in 1955 there was one RCA Synthesizer, today there are millions of systems of equal power.

REFERENCES

1. Helmholtz, Hermann L. F., **On the Sensations of Tone,** translated from the fourth German edition of 1977 by Alexander J. Ellis (first edition, 1862), Dover, N.Y. (1954).

2. Baker, Ray Stannard, "New Music for an Old World", *McClures*,pp. 291-301 (1906).

3. Rhea, Thomas, "The Evolution of Electronic Musical Instruments in the United States", Ph. D. dissertation, George Peabody College for Teachers, University Microfilms, Ann Arbor, MI (1972).

4. Rhea, Thomas,"The History of Electronic Musical Instruments", **The History of Electronic Musical Instruments**, G. Armbruster, ed, Quill, N.Y.(1984).

5. Weidenaar, Reynold, "The Telharmonium: A History of the First Music Synthesizer, 1893-1918", Ph. D. dissertation, New York University (1988).

6. Miessner, Benjamin F., "Electronic Music and Instruments", *Proc. I.R.E.*, Vol. 24, No. 11, pp. 1427-1463 (1936).

7. LeCaine, Hugh, "Electronic Music", *Proc. I.R.E.*, Vol. 44, No. 4, pp. 447-478 (1956).

8. Rockmore, Clara, "The Art of the Theremin", D/CD 1014, Delos International, Santa Monica, CA (1987).

9. Trautwein, Frederich, "Elektrishche Musik", Verlag Weidman, Berlin (1930).

10 Rhea, Thomas, "the ondes martenot, an early milestone in the development of electronic keyboards", *Keyboard*, p. 14 (June, 1984).

11. Oskar Sala, "Mixture-Trautonium and Studio Technique", *Gravesaner Blatter*, Vol. 6, pp. 53-60 (1962).

12. Dorf, Richard H., "Hammond Organs", in **Electronic Musical Instruments**, Radiofile, N.Y. (1968).

13. Olson, Harry F. and Belar, Herbert, "Electronic Music Synthesizer", *J. Acoust. Soc. Am.*, Vol. 27, No. 3, pp. 595-612 (1955).

14. Luening, Otto, "Origins" in **Development and Practice of Electronic Music,** Appleton and Perera, eds, pp. 1-21, Prentice-Hall (1975).

15. Cross, Lowell, "Electronic Music, 1948-1953", *Perspectives of New Music*, Vol. 6, pp. 32 - 65 (1968).

16. Schwartz, Elliott, **Electronic Music, A Listener's Guide**, Praeger, N.Y.(1975).

17. Manning, Peter, **Electronic and Computer Music**, Clarendon Press, Oxford (1987).

18. Ciamaga, Gustav, "The Tape Studio" in **Development and Practice of Electronic Music**, Appleton and Perra, eds, ,pp. 68-137, Prentice Hall (1975).

19. Moog, Robert A., "Voltage-Controlled Electronic Music Modules", *J. Audio Engr. Soc.*, Vol. 13, No. 3, pp. 200-206 (1965).

20. Carlos, W. and Folkman, B., "Multi-Track Recording in Electronic Music", *Electronic Music Review*, No. 6, pp. 20-32 (April, 1968).

21. Beauchamp, James, "The Harmonic Tone Generator, a Voltage-Controlled Device for Additive Synthesis of Harmonic Spectra", *Audio Engr. Soc. Preprint* No.323 (1964). Also, "Additive Synthesis of Harmonic Musical Tones", *J. Audio Engr. Soc.*, Vol. 14, pp. 332-342 (1966).

22. Moog, Robert A., "Electronic Music -- Its Composition and Performance", *Electronics World* (Feb., 1967).

23. Mathews, Max V., "An Acoustical Compiler for Music and Psychological Stimuli", *The Bell System Technical J.*, Vol. 40, pp. 677-694 (May, 1961).

24. Mathews, Max V., "The Digital Computer as a Musical Instrument", *Science*, Vol. 142, pp. 553-557 (Nov., 1963).

25. Roads, Curtis, "Interview with Max Mathews", *Computer Music J.*, Vol. 4, No. 4, pp. 15-22 (1980).

26. von Foerster, Heinz and Beauchamp, James, eds, **Music by Computers**, Wiley & Sons (1969).

27. Mathews, Max V., **The Technology of Computer Music**, M.I.T. Press (1969).

28. Dodge, Charles and Jerse, Thomas, **Computer Music: Synthesis, Composition, and Performance**, Schirmer Books (1985).

29. Moore, F. Richard, **Elements of Computer Music**, Prentice Hall (1990).

30. Chamberlin, Hal, "Delayed Playback Music Synthesis Using Small Computers", *Proc. Symposium on Small Computers in the Arts*, pp. 27-32 (1981).

31. Chowning, John, "The Synthesis of Complex Audio Spectra by Means of Frequency Modulation", *J. Audio Engr. Soc.*, Vol. 7, No. 7, pp. 526-534 (1973).

32 Buxton, William A. S., "A Composer's Introduction to Computer Music", *Interface*, Vol. 6, pp. 57-72 (1977).

33. Alonso, Sydney, Appleton, Jon, and Jones, Cameron, "A Special Purpose Digital System for Musical Instruction, Composition, and Performance", *Computers and the Humanities*, Vol. 10, pp. 209-215 (1976).

34. Smith, Julius O., "Physical Modeling Using Digital Waveguides", *Computer Music J.*, Vol. 16, No. 4, pp. 74-91 (1992).

35. Samson, Peter, "A General-Purpose Digital Synthesizer", *J. Audio Engr. Soc.*, Vol. 28, No. 3, pp. 106-113 (1980).

36. Alles, H.G. and di Giugno, Pepino, "A One Card 64-Channel Digital Synthesizer", *Computer Music J.*, Vol. 1, No. 4, pp. 7-9 (1977).

37. Alles, Harold G., "Musical Synthesis Using Real Time Digital Techniques", *Proc. IEEE*, Vol. 68, No. 4, pp. 436-449 (1980).

38. Wallraff, Dean, "The DMX-1000 Signal Processing Computer", *Computer Music J.*, Vol. 3, No. 4, pp. 44-49 (1979).

39. Truax, Barry, "Real-Time Granular Synthesis with a Digital Signal Processing Computer", *Computer Music J.*, Vol. 12, No. 2, pp. 14-26 (1988).

40. Haken, Lippold and Hebel, Kurt, "The Platypus Programmers' Reference Manual", internal report, Computer-based Education Research Laboratory, University of Illinois at Urbana-Champaign (1987).

41. Scaletti, Carla, "The Kyma/Platypus Computer Music Workstation", *Computer Music J.*, Vol. 13, No. 2, pp. 23-38 (1989).

42. Miller, Dennis, "Symbolic Sound Kyma System 4.0 (Win, Mac): A synthesis workstation with unlimited sounds potential", *Electronic Musician*, Vol. 11, No. 7 (July, 1995).

43. Jaffe, David, "Overview of the NeXT Music Kit", *Proc. 1989 Int. Computer Music Conf.*, pp. 135-138 (1989).

44. Wiggen, Knut, "Electronic Music Studio at Stockholm, its Development and Construction", *Interface*, Vol. 1, pp. 127-165 (1972).

45. Mathews, Max and Moore, F. Richard, "GROOVE--A Program to Compose, Store, and Edit Functions of Time", *Comm. ACM*, Vol. 13, No. 12, pp. 715-721 (1970).
Also, Moore, F. R., "Computer Controlled Analog Synthesizers", Bell Telephone

Laboratories internal report (c. 1970).

46. Zinovieff, Peter, "A Computerized Electronic Music Studio", *Electronic Music Reports*, No. 1, pp. 5-22 (1969).

47. Buxton, William et al, "An Introduction to the SSSP Digital Synthesizer", *Computer Music J.*, Vol. 2, No. 4, pp. 28-38 (1978).

48. Beauchamp, James; Pohlmann, Ken; and Chapman, Lee, "The TI980A Computer-Controlled Synthesis", *Proc. 1975 Int. Computer Music Conf.* (Second Annual Music Computation Conf.), J. Beauchamp and J. Melby, eds, Vol. 1, pp 1-28 , Computer Music Assn., San Francisco (1976).

49. Murray, David, Beauchamp, James, and Loitz, Gary, "Using the PLACOMP/TI980A Music Synthesis System: The PLACOMP Language", *Proc. 1978 Int. Computer Music Conf.*, C. Roads, ed, Vol. 1, pp. 151-166, Computer Music Assn., San Francisco (1979).

50. Schmid, Valerie and Haken, Lippold, "The Interactive Music System User's Manual", CERL Music Group, 252 Engineering Research Lab, UIUC (1983).

51. Scaletti, Carla, "The CERL Music Project at the University of Illinois", *Computer Music J.*, Vol. 9, No. 1, pp. 45-58 (1985).

52. Rona, Jeffrey, "A Recording Engineer's Guide to MIDI", *Recording-engineer/Producer*, pp. 125-129 (December, 1983).

53. Loy, Gareth, "Musicians Make a Standard: The MIDI Phenomenon", *Computer Music J.*, Vol. 9, No. 4, pp. 8-26 (1985).

54. Moog, Bob, "MIDI: Musical Instrument Digital Interface", *J. Audio Engr. Soc.*, Vol. 34, No. 5, pp. 395-404 (1986).

55. "MIDI 1.0 Detailed Specification" (this has been continually updated since the original version was published in 1984 by Sequential Circuits, Inc.), The International MIDI Association, Los Angeles, CA (1984 -).

56. "Standard MIDI Files 1.0", The International MIDI Association, Los Angeles, CA (1988).

ECE 302 Notes Chapter Two

PARAMETERS OF MUSICAL EXPRESSION

Contents

PARAMETERS OF MUSICAL EXPRESSION

2.0 Introduction

Musicians and composers continually work with four aspects of music -- **pitch, loudness**(dynamic), **timing**, and **timbre**. Over the years, they have evolved complex methods of notation, composition, and performance as components of a musical culture, which has been transferred from teacher to student over successive generations.

It can be tempting to assume that black notes on the staves of a musical score precisely define a musical performance. In fact, these are only "indicators" for performers to follow. While the frequencies and durations played must indeed fall within certain boundaries, the parameters are usually performed with much less exactness than we might expect. In general, a great deal of "interpretation" of the score is required from musicians for them to produce desireable results. When scoring for traditional instruments or voices, a composer must express the sounds he imagines in a notation which takes into account the limitations of acoustic instruments and their performers to create a rendition which satisfies both the composer and the audience.

Musical parameters are varied according to the context of the music performed. Musicians recognize the concept of musical hierarchy -- that notes are contained in phrases, phrases are contained in themes, and themes are contained within larger musical structures. Moreover, much music has dramatic content; frequently a long piece will spell out a "story", even though no real text is involved. For example, a simple abstract story could be: STORM, CALM, BUILDING TENSION, EXPLOSION, CALM. These contextual situations can strongly influence the ways notes are performed within various sections of a piece.

The limitations of acoustic instruments do not hold for electronic music. For example, it is easily within the grasp of technology to provide any pitch tuning system, any rhythmic sequence, and any set of waveforms.

In the case of delayed-performance electronic music, composers may not have (or want) performers to "interpret" their instructions. It follows that composers will be responsible for many aspects of music which they formerly would have left to performers. Unless they themselves perform (i.e., into a processing machine), they must select "values" for amplitude, frequency, duration, and subcomponents of timbre. Musical parameters are "control parameters" for the music-making apparatus. The distinction between the composer's responsibilty for parameter definition in electronic music and what he would designate in traditional music is particularly evident for the dimension of timbre. In traditional music, once the instrument type is chosen, the musician, rather than the composer, assumes complete responsibility for the timbral outcome. In electronic music, the composer usually has a wide choice of sounds (within the limits of his equipment) at any given moment of time, and frequently must decide on many details which determine the overall qualities of those sounds.

In the case of real time electronic performance, many nuances which we take for granted in acoustic instrument sounds, which result from mechanical interfaces between a player's lips or fingers and the parts of the instrument he contacts, must be consciously simulated in an electronic instrument. An

engineer who designs equipment for musicians or composers should strive to provide as much variability and precision of control of various parameters as is practically possible, keeping in mind that there needs to be a strong relation between performance control and the sonic result. At the same time there needs to be elements in the sound which provide variety and, paradoxically, are beyond the control of the musician.

Over the years our ears have evolved, and our expectations about what is to be heard have become specialized. We have developed at least four modes of listening: speech, music, noise, and background. Each mode is distinguished by its typical sounds, and, especially in the cases of speech and music, there is considerable information to be processed. We might say that speech and music both carry information based on parameter variations within certain limited expected ranges. Style of performance is one (rather subtle) manifestation of parameter variations, which an experienced musical listener can easily detect. A more obvious listener task is to detect which instrument(s) are currently being played in a piece of music.

To begin to understand musical parameters from some sort of systematic perspective, we can examine them from the complementary viewpoints of a musicologist, a physicist, and a psychologist. In music, it is important to consider how sounds affect a listener in musical situations -- within a piece of music. In physics it is important to measure certain physical parameters (frequency, amplitude, time, sound spectrum) as they are performed by musical instruments. In psychoacoustics (sound perception) it is important to understand how listeners react to various sound stimuli, although usually not in a musical context. Information about performed music may be gleaned from recordings and scores. On the other hand, information about the physics of music and the perception of sound is contained in various published papers and books. A familiarity with all three approaches can provide a good background to aid the design of electronic musical instruments and devices.

To contrast the three approaches, music, physics, and psychoacoustics, we might look briefly at how the four principal parameters could be expressed in specific cases:

	MUSIC	PHYSICS	PSYCHOACOUSTICS
1.	middle C (C4) (note, pitch, or tone)	261.6 Hz (frequency)	350 mels (pitch)
2.	mf (dynamic)	75 dB SPL* (amplitude, intensity, or level)	15 sones* or 75 phons* (loudness or loudness level)
3.	quarter note (duration, fraction of beat, tempo)	0.2 sec* (time duration)	? (interval)
4.	clarinet @F4, mf	500,18,65,100,60,15, 40,18,35,41,10,14,10* @f_1 = 358 Hz	"hollow" (a semantic label)
	(instrument tone)	(harmonic spectrum)	(timbre)

* typical values

Each discipline has its own vocabulary and its own measurement or description system for each of the four parameters. It is possible to obtain a kind of rough translation between the three interpretations of each parameter, but we cannot expect an exact one-to-one correspondence.

Given the above discussion, one might wonder about such questions as "Is it possible to notate music with absolute precision?" or "Is it possible to notate 'great music' in terms of a sequence of symbols?". While we have no definitive answers to these questions, we note that one must start somewhere, and that it can be fascinating to explore the connection between music, science, and technology. In the coming sections we will discuss 1) pitch and frequency, 2) loudness and amplitude, 3) duration and timing control in music, and 4) coding schemes for music. A discussion of timbre and its analysis and synthesis is left to Chapter 3.

2.1 Pitch and Frequency

In ancient Greece as early as 500 B.C. Pythagoras and his followers recognized the strong connection between musical pitch and the lengths of vibrating strings. However, the actual relationship between vibration frequency and pitch was not demonstrated until the sixteenth century by scientists like Galileo and Mersenne. An accurate definition of frequency required a method for precise measurement of time, which in turn required the invention of the pendulum. Pitches associated with well known sound sources could be matched with frequencies generated by sirens. These are easy to calibrate, since the frequency of a siren is quite obviously the product of its rotational rate (e.g., 1/sec) and the number of holes around its circumference.

On the other hand, pitch and frequency are not, strictly speaking, synonymous terms. Pitch refers to a particular perception and can not be directly observed. Frequency can be measured by electronic apparatus. The American National Standards Institute defines pitch as "that attribute of auditory sensation in terms of which sounds may be ordered on a scale extending from low to high". It seems clear from this statement that one must have experienced a pitch sensation in order to understand this definition. The ANSI further states that "The pitch of a sound may be described by the frequency or frequency level of that pure (sine) tone having a specified sound level that is judged by subjects to produce the same pitch." In other words, the sine tone is the signal that should be used for pitch comparisons. This is in spite of the fact that isolated sine tones seldom occur in in music. In the symphony orchestra an oboe tone, a tone which is unusually rich in harmonic overtones, is usually used for tuning the other instruments.

Researchers have demonstrated that pitch depends not only on frequency but also on amplitude, waveform, and duration. For example, Stevens observed [1935] that pitches of sine tones above 3000 Hz rise with increasing intensity, whereas pitches of tones below 1000 Hz fall under similar circumstances. Sounds which do not have a definite period may still have a definite pitch, and sounds which are periodic but lack the fundamental component or even the first few harmonics still usually have a pitch corresponding to the fundamental. The complexities of pitch perception and of musical sound spectra have made it difficult to design automatic pitch detectors, even for solo instrument recordings. Automatic transcription of music for a wide variety of acoustical instruments and over wide frequency ranges is a difficult problem, although some very good partial solutions are now available for solo instruments.

In music there are "absolute pitches" and "relative pitch intervals". A melody consisting of a series of pitches is invariant with respect to transposition to a new key. In fact, most people cannot tell what key (absolute pitch) the melody is being played in, but they are able to instantly recognize the melody. This

recognition is based on a particular pattern of pitch intervals, which corresponds to particular set of frequency ratios. It is no wonder that the diatonic and chromatic scales were arrived at with much greater certainty than were the absolute pitch standards. The frequency of A4 varied from 377 Hz in the 14th century to as high as 567 Hz during the 19th century [Ellis, 1885]. This was settled by an international conference held in London in May, 1939, which unanimously adopted the frequency 440 Hz as the standard for the treble cleff note A4 [Wood, 1944].

2.1.1 Scales and Tuning

The equal-tempered tuning system came into use sometime during the 18th century and by the mid-19th century was adopted as a "standard" for pipe organs and pianos [Ellis, 1885]. Nevertheless, string, wind, and vocal performers have always been free to vary their pitches at will, so there is no guarantee that any particular tuning will actually be used in a live performance.

While A4 (440 Hz) is considered the standard reference pitch, AO (27.5 Hz), the "lowest octave A" can be considered the basis for pitch-frequency calculations. Also, for historical reasons octave numbers change between B and the next higher note C (e.g., B3 is followed by C4) instead of G and A, as might be expected. In the **equal-tempered** system, all adjacent notes within an octave are related by ratios of the twelth root of two. C being nine chromatic notes below A, the frequency of middle C is

$$f_{C4} = 2^{-9/12} \times 440 = 261.626 \text{ Hz}$$

and in the lowest octave the first note C0 has a frequency of

$$f_{C0} = f_{C4}/2^4 = 16.352 \text{ Hz} = f_0$$

Within each octave the pitches are numbered as follows:

	C	C#	D	D#	E	F	F#	G	G#	A	A#	B	C
p =	0	1	2	3	4	5	6	7	8	9	10	11	12 (or 0)

Then, given the octave O and pitch number p for a pitch, its frequency can be calculated using the formula

$$f_{O.p} = 16.352 \cdot \times 2^{(O + p/12)} \qquad [2.1.1]$$

Inversely, we can obtain the (absolute) pitch value from the frequency using the formulas

$$O.p = 12 \bullet \log_2(f/f_0) \qquad \text{(total absolute pitch)}$$

$$O = \text{int}[\log_2(f/f_0)] \qquad \text{(the octave)}$$

$$p = 12 \bullet \text{frac}[\log_2(f/f_0)] \qquad \text{(pitch within octave)}$$

where $f_0 = f_{0.0} = 16.352 \text{ Hz} = f_{C0}$.

p can lie anywhere between two integer values. Pitches that lie between integral values of p are sometimes referred to as microtones.

Measurement of frequency in **cents** is quite often used in music theory, particularly to indicate small tuning departures. There are 100 cents in one semi-tone, i.e., corresponding to one p unit. We can define absolute cents as follows:

$$\text{cents\_abs} = 1200 \cdot \log_2(f/f_0) = 1200 \cdot \log_2(2^{O+p/12}) = 1200 \cdot O + 100p \qquad [2.1.2]$$

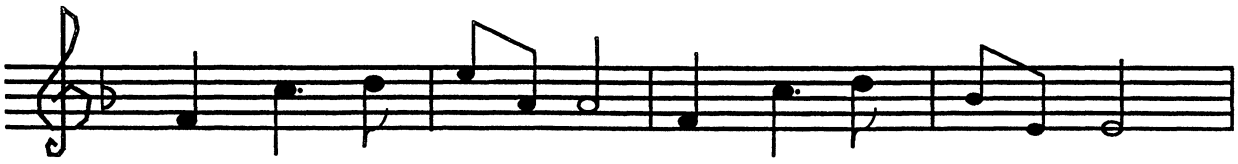
For example, if we choose a frequency f = 700 Hz we get $O = 5$, p = 5.04, and cents_abs = 6504.

However, the real value in using p or cents calculations is for the case of pitch or frequency intervals *relative* to some reference pitch or frequency besides f_0. If a frequency ratio is given by f_2/f_1, the pitch difference is $\Delta p = 12 \log_2(f_2/f_1) = p_2 - p_1$ and $\Delta\text{cents} = 100\,\Delta p$.

To aid in calculating $\log_2(\,)$, note that $\log_2(x) = 1.442695 \log_e(x) = 3.321928 \log_{10}(x)$. Thus, the formula for computing cents (relative or absolute) can be written

$$\Delta\text{cents} = 1200 \log_2(f_2/f_1) = 1731.2 \log_e(f_2/f_1) = 3986.3 \log_{10}(f_2/f_1)$$

Here is an example applied to a melodic sequence of notes:



f =	349.2	523.2	587.3	659.3	440.0	349.2	523.3	587.3	466.2	329.6
f/f_{ref} =	1.000	1.498	1.682	1.888	1.260	1.000	1.498	1.682	1.335	0.994
Δp =	0	7	9	11	4	0	7	9	5	-1
Δcents =	0	700	900	1100	400	0	700	900	500	-100

In this case the frequency ratios are fairly complex, but the relative pitch values are simple integers.

An alternative tuning system is **just intonation**, which is based on the concept of simple integer frequency ratios, i.e., the original notion of the Greeks that consonant (harmonious) sounding intervals are produced by two strings which are identical except for their lengths and which are in simple integer ratios. The frequency ratios in order of consonance, together with the corresponding pitch values are given as

frequency ratio	1:1 (unison)	1:2 (octave)	2:3 (fifth)	3:4 (fourth)	4:5 (major third)	5:6 (minor third)	8:9 (major second)	15:16 (minor second)
Δcents	0	1200	702	498	386	316	204	112

major second = whole step
minor second = half step

While the just frequency ratios are simple, their corresponding relative pitch values are more complex. By comparing the Δcents values of the just intervals with their equal-tempered equivalents we can see exactly how much the differences between the two are. The worst cases are for the major and minor thirds which are off by 14 and 16 cents, respectively. However, the fifth and fourth intervals are only off by 2 cents and the whole step is only off by 4 cents. Thus, except for the thirds, the equal-tempered scale is a reasonable approximation to the just scale.

The just scale ratios and equivalent cents values are given below:

note	C	D	E	F	G	A	B	C
ratio	1	9/8	5/4	4/3	3/2	5/3	15/8	2
cents	0	204	386	498	702	884	1088	12

One of the interesting properties of this scale is that each major triad (CEG, FAC, and GBD) has a perfect ratio of 4:5:6; also, two of the minor triads have "perfect" ratios of 10:12:15 and one of the minor triads is "imperfect" (the reader is invited to figure out which one). However, one note of the imperfect minor triad can be temporarily adjusted to make it perfect.

Harmonic frequencies are the basis of the just scale. Given the fundamental frequency of a tone, f_1, the frequency of harmonic k is given by

$$f_k = k f_1 \qquad [2.1.3]$$

Harmonics also have a definite cents relationship with respect to one another. Relative to the fundamental frequency, we can calculate the harmonic intervals in cents using

$$\text{cents}_k = 3986.3 \log_{10}(f_k/f_1) = 3986.3 \log_{10}(k) \qquad [2.1.4]$$

It has often been stated that just intonation is superior to equal-tempered. It is true that perfect ratios tend to eliminate beats between harmonics of different notes in chords[c.f., Helmholtz, Chap. 16,17].. For example, we note that for a C major just scale the third harmonic of the C agrees perfectly with the second harmonic of the G. Likewise the fifth harmonic of the C is exactly equal to the fundamental of the E. Thus, when the C-E-G triad is played, there is a minimum number of distinct frequencies and a mininum of beating between frequencies. But can we really hear the difference between a just-tuned and equal-tempered version of a composition? Our experience is that for some people (notably musicians) the difference is obvious,but for others it is very subtle. Nevertheless, most people can hear a difference if they listen carefully enough. Just tuned harmonies have a certain "purity" lacking with equal-tempered tuning. In performed music, just tuning occurs particularly in cadences with instruments or voices where little vibrato is used (e.g., barber shop quartets or boy's choirs).

Keyboards tuned to the just system run into problems when attempts are made to modulate from one key to another. In fact, the equal-tempered system was the solution to the historical problem of modulation on fixed-tuned instruments. Given a just-tuned C scale, if we wish to create just-tuned scales based on the neighboring keys of G and F, we must introduce two new pitches for each scale. This becomes clear if we look at the ratios of adjacent notes of the just scale:

note	C	D	E	F	G	A	B	C
ratio	1	9/8	5/4	4/3	3/2	5/3	15/8	2
	\	/ \	/ \	/ \	/ \	/ \	/ \	/
step ratio		9/8	10/9	16/15	9/8	10/9	9/8	16/15

Whereas there is only one half-step ratio, there are two different whole step ratios. This unique succession of intervals gives the just scale a *signature*, which identifies C as the first note of the scale. If we now construct a just scale starting on G, we arrive at the following pattern:

note	G	A	B	C	D	E	F#	G
ratio	3/2	27/16	15/8	2	9/4	5/2	45/16	3
	\	/ \	/ \	/ \	/ \	/ \	/ \	/
step ratio		9/8	10/9	16/15	9/8	10/9	9/8	16/15

The pitch of G does not change, but since the ratio G to A is now 9/8 instead of 10/9, the pitch of A's ratio must change. Working out the remaining ratios, we find that the pitches of B, C, D, and E have not changed (except for rising an octave), but, of course, F# is a new pitch. Thus, two new tunings are required for the key of G.

It is interesting to calculate the ratio of the new A to that which it had in the C scale. This is a ratio that keeps popping up in scale theory and is called the **Syntonic Comma**.

$$SC\_ratio = (27/16)/(5/3) = 81/80 = 1.0125$$

$$SC\_cents = 21.5$$

This corresponds to about one-fifth of a semi-tone.

If we base a just-tuned scale on the pitch of F, we will arrive at the same conclusion: All of the ratios are the same, except for two new ones. The reader can demonstrate that the two new ratios are Bb at 16/9 and D at 10/9.

In conclusion, for the just scale, modulation to the next lower neighboring key (down a fourth) introduces the ratios 16/9 and 10/9, replacing the former 15/8 and 9/8. Modulation to the next higher neighboring key (up a fifth) introduces the ratios 45/32 and 27/16. If we start at the key of Gb and modulate to higher and higher neighboring keys, traversing through the keys of Db, Ab, Eb, Bb, F, C, G ..., we eventually will arrive at the key of F#, which is considered to be **enharmonic** to the key of Gb. This is called traversing through the "circle of fifths", and the pitch ratio we arrive at is nearly equal to that of the original Gb, after we divide out all of the octaves. This magic ratio is called the **Pythagorean Comma** and is given by

$$Pyth\_ratio = 3^{12}/2^{19} = 1.0136$$

$$Pyth\_cents = 23.5$$

The Pythagorean comma is only 2 cents different than the Syntonic Comma, a barely noticeable difference. It follows that if we have just tunings for all of the chromatic keys, we will have covered all the modulation possibilities.

2.1.2 The Smallest Perceptible Frequency Differences

How accurate must pitches be? More precisely, given two tones having the same amplitude, duration, and waveform and frequencies f_a and f_b, what is the smallest difference in the two frequencies (f_a - f_b) which can be reliably heard? In an effort to answer this question psychoacousticians have performed systematic tests on human subjects. The answer depends on the exact way the test is performed.

In 1931 Shower and Biddolph published the results of their (now classic) test of the frequency **just noticeable difference** (JND) [Shower and Biddolph, 1931]. Their method was to frequency modulate sine tones of various frequencies at a rate of around 2 Hz and vary the amount of modulation. They found that the amount of modulation necessary for human detection varied with the amplitude as well as the frequency of the sine tone. However, for tones above 40 dB IL (defined in the next section) the dependence on amplitude was slight. For sine tones the dependence of Δf_{JND} on frequency can be summarized as follows:

1. Between 62.5 Hz and 1000 (4 octaves) Δf_{JND} varies between 2 to 3 Hz.
2. Above 1000 Hz Δf_{JND} is almost a constant fraction of frequency with
   $.003\, f < \Delta f_{JND} < .005\, f.$

The Shower-Biddolph experiment has been repeated by other researchers using variations on their methods. Except for some details, the basic results of the original researchers have been confirmed. We can interpret from these results that for sine tone pitches above C6 (1046 Hz) frequencies must be accurate within 0.5 % or about 8.5 cents (about one twelfth of a semitone). However, below this pitch greater and greater percentage mistunings can be tolerated. For example, at C2 (65.4 Hz) a 2 Hz mistuning is barely noticeable, corresponding to 3.1% or 0.5 semitone! This is not exactly what a musical listener would expect, but remember, the test tones are sine waves.

In real musical situations sine tones are quite rare. Most musical waveforms are more complex and contain higher frequencies (harmonics) within the tones. Henning and Grosberg [1968] measured Δf_{JND} for pulse waves containing harmonics up to the audible limit (20 KHz) and found that pitch discrimination of these sounds was much improved over that of sine waves. Researchers at Bell Laboratories [Flanagan and Saslow, 1958 and Klatt, 1973] found that Δf_{JND} for synthethic speech vowels was around 0.3 Hz. For typical male voices, the fundamental frequencies are generally around 100 - 150 Hz and have low amplitudes compared to the upper harmonics. Think about it this way: If a 100 Hz fundamental varies by .3 Hz, its tenth harmonic varies by 3 Hz, and at 1000 Hz this is enough to be barely perceptible. This seems to be a sufficient explanation as to why complex low pitched tones require more careful tuning than sine tones of the same frequency.

In summary, it is fair to say that for complex tones (those with several harmonics) 0.2% tuning accuracy is needed for all pitches within a wide range. The absolute limit of detection may be even lower. We might be able to take a cue from tape recorder wow and flutter specifications. Figures as low as .01% have been quoted. However, there are relatively few electronic instruments which can measure frequency changes of so small a magnitude.

2.2 Dynamics, Loudness, Intensity Level, and Amplitude

2.2.1 Calculation of Intensity Levels, Loudness Levels, and Loudnesses

Physicists and psychoacousticians have worked out concrete methods of measuring sound amplitude in physical and subjective terms. They have devised methods to predict (with limited accuracy) the loudness of several sounds, whose individual intensities and loudnesses are known, combined into a single sound. First of all, we must define sound **intensity** in terms of **amplitude.**

$$I = \gamma A^2 \quad (\text{watts/m}^2) \qquad [2.2.1]$$

where A corresponds to some "linear" physical quantity (e.g., volts, amperes, pressure units such as Newtons/m^2, etc.) and γ is a constant depending on the system of measurement.

Sound intensities occur over an extremely wide range of values:

$I = 10^{-12}$ watts/m^2 (barely audible under the best of circumstances)

$I = 10^{-4}$ watts/m^2 (moderately loud sound)

$I = 1.0$ watt/m^2 (barely tolerable loud sound)

$I = 2.5\times10^{7}$ watt/m^2 (intensity corresponding to atmospheric pressure)

For this reason a logarithmic measure is convenient. Intensity Level IL (sometimes called sound intensity level SIL) is defined as

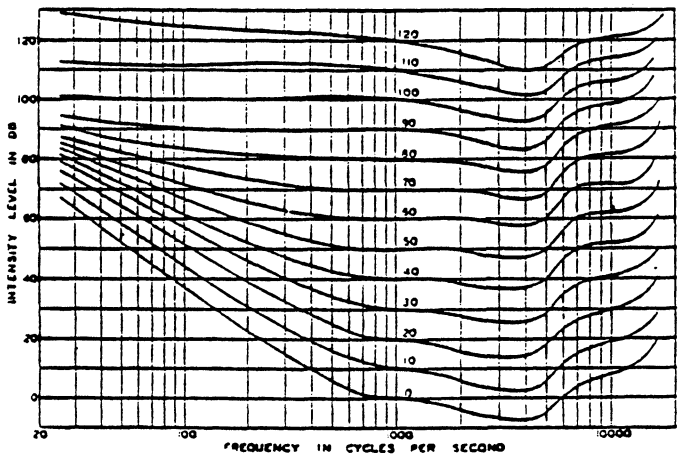
$$IL = 10 \log_{10}(I/I_{ref}) \quad (\text{dB}) \qquad [2.2.2]$$

where $I_{ref} = 10^{-12}$ is the reference intensity corresponding to 0 dB.

Life would be simpler if all frequencies (of sine tones) sounded equally loud, but such is not the case. During the 1930's Fletcher and Munson tabulated the subjective loudness judgements of a great number of listeners in an effort to determine data appropriate for the design of telephone systems [Fletcher and Munson, 1937]. One result of their work is the set of curves shown in Figure 2.1a, which are generally referred to as the "Fletcher-Munson" curves or "equal loudness contours". For any pair of values (frequency, Intensity Level), one can quickly determine the corresponding **Loudness Level** (in **phons**). The Loudness Level of a sound may be defined as follows:

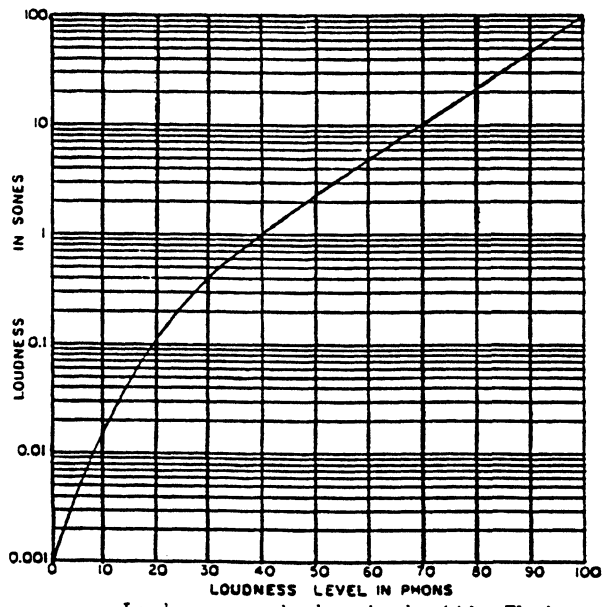
> **The Loudness Level of a sound is the intensity level of a 1000 Hz sine tone which sounds equally loud as the sound in question.**

Thus we see, according to Figure 2.1a, that a 100 Hz sine tone having an Intensity Level of 60 dB (at the entrance to the ear canal) has a Loudness Level of 37 phons. Also, note that by definition the Loudness Level and Intensity Level of a 1000 Hz sine tone are exactly the same .



Contour lines of equal loudness for normal ears. Numbers on curves indicate loudness level in phons. 0 decibels = 0.000204 dyne per square centimeter. (*After Fletcher and Munson.*)

(a)



Loudness versus loudness level. (*After Fletcher.*)

(b)

Figure 2.1 a) Intensity Level vs. Frequency for Equal Loudness of Sine Tones.
b) Loudness (in Sones) vs. Loudness Level in Phons

Some general observations can be made about the equal loudness contours. First, for low Intensity Levels the contours curve sharply upward at low frequencies, so that the range of Loudness Levels corresponds to a relatively compressed range of Intensity Levels. However, at high Intensity Levels the curves are practically flat. The lowest contour, corresponding to 0 phons, is also called the **threshold of hearing** contour. Presumably, no sine tone is audible below the 0 phon level. Note the "dip" in all of the contours in the range 2000 to 5000 Hz. This is due to a principal resonance of the ear canal occuring at about 3000 Hz, which results in the ear being more sensitive in this region. Frequencies above 6000 Hz become harder to hear as is indicated by all contours.

However, these curves are actually for "young normal ears". Hearing damage and/or natural aging can alter the curves significantly. A measurement of the threshold of hearing contour is a sharp indicator of hearing loss. This occurs if one's threshold curve lies significantly higher than the "normal curve" for his age. While the threshold curve will lie higher on the chart with increasing age, there should be no unusual peaks on the curve. A peak could indicate a hearing loss in a specific range of frequencies.

We can infer from Figure 2.1a that if the Intensity Level and frequency of a sine tone are within certain limits (called the "mid-band" region), we can approximate

$$\text{LL (phons)} \approx \text{IL (dB)} \quad \text{for} \quad \begin{array}{l} \text{IL} > 40 \text{ dB} \\ 500 < f < 5000 \text{ Hz} \end{array} \qquad [2.2.3]$$

Since the frequency components of many sounds are mainly in mid-band, Loudness Level and Intensity Level are often approximately the same. In general, we can think of Loudness Level as *equalized* Intensity Level.

The equal loudness contours were determined by attempting to equate the loudnesses of various frequency sine tones with certain levels of 1000 Hz tones. But there was nothing which guaranteed that equally-spaced LL's would be perceived as equally-spaced in loudness -- i.e., that 80 phons was to 70 phons as 70 phons was to 60 phons, etc. Consequently, Fletcher and Munson determined [Fletcher & Munson, ibid.] another curve, which gives a scale for subjective Loudness (in **sones**) as a function of Loudness Level.

The sones Loudness scale was determined by the method of "doubling", where subjects (listeners) were asked to repeatedly determine which Loudness Level of a sound B sounded "double in loudness" compared to the Loudness of sound A, which was at a known, fixed Loudness level. The result of these tests is shown in Figure 2.1b. The usual starting point on this graph is at LL=40 phons, where 1.0 sones is the Loudness by definition. Increasing to 49 phons, we see that the loudness has doubled to 2.0 sones; increasing to 58 phons, we get L=4.0 sones; and so on.

The result is that for LL > 40 dB, we can use the simple approximation:

$$\text{L (sones)} \approx 2^{(\text{LL}-40)/9} \qquad [2.2.4]$$

For LL's below 40 phons, the curve is much steeper. Taking the example given above of a 100 Hz sine tone having an intensity level of 60 dB, where its Loudness Level was determined to be 37 phons, we see that according to Fig. 2.2b, its Loudness is 0.7 sones.

The real value of the Loudness-vs.-Loudness Level graph is not for its ability to give loudnesses in sones,

but for its use in determining the total Loudness Level of a combination of sine tones at different frequencies. This is because it has been determined that the total Loudness of a sound containing several *distinctly different* frequencies is simply the sum of the Loudnesses of the individual components.

Let a sound consist of several distinct sine waves having frequencies and Intensity Levels, $(f_1, IL_1), (f_2, IL_2), (f_3, IL_3), \ldots$ First, we use the equal loudness contours to determine the set of level graph levels $LL_1, LL_2, LL_3, \ldots$ Next,we use the Loudness-vs.-Loudness Level graph to obtain $L_1, L_2, L_3 \ldots$. The total Loudness of the sound is given by

$$L_{tot} = L_1 + L_2 + L_3 + L_4 + \ldots \quad \text{(sones)}$$

From L_{tot} we can then determine the equivalent Loudness Level by using Figure 2.1b or Equation 2.2.4 in the inverse manner (e.g., 10 sones total corresponds to 70 phons total). This total Loudness Level (LL_{tot}) then corresponds to the Intensity Level of a 1000 Hz sine tone which matches the total aggregate sound in judged loudness. We can call this the *loudness equivalent intensity level*:

$$IL_{tot} = LL_{tot} = 40 + 9 \log_2(L_{tot}) = 40 + 29.9 \log_{10}(L_{tot}) \qquad [2.2.6]$$

While this procedure does not give absolutely consistent results, it does give reasonably good estimates. The procedure can be used, for example, to determine the Loudness of a complex waveform given its fundamental frequency, its overall Intensity Level, and its Fourier amplitudes as follows:

Suppose we have the Fourier amplitudes $A_1, A_2, A_3, \ldots$ and the total Intensity Level IL_{tot}. The intensity I_{tot} is given by

$$I_{tot} = 10^{12}\, 10^{IL_{tot}/10} \quad (\text{Watts/m}^2) \qquad [2.2.7]$$

Corresponding to

$$I_{tot} = I_1 + I_2 + I_3 + \ldots, \qquad [2.2.8a]$$

the total squared amplitude of the waveform is given by

$$A_{tot}^2 = A_1^2 + A_2^2 + A_3^2 + \ldots \qquad [2.2.8b]$$

and $I = \gamma A_{tot}^2$. This should determine γ. Next, we find each I_k using $I_k = \gamma A_k^2$ and the corresponding intensity levels from

$$IL_k = 10 \log_{10}(I_k/10^{-12}) \qquad [2.2.9]$$

From the IL_k values we use the equal-loudness contours to determine the set of LL_k and the Loudness-vs.-Loudness Level curve to determine the set of L_k. Finally, we add the L_k to get L_{tot} and, again using the L-vs.-LL curve, convert back to LL_{tot}. This is the final answer.

Strictly speaking, this technique only works for components whose frequencies are "distinctly different.".

If the frequencies are close together, we should compute the total loudness from the total intensity translated into loudness level using Figure 2.1b or Equation 2.2.4. What are the criteria for frequencies to be "distinctly different"? This turns out to be a complicating factor, which depends on such properties of our ears as "critical bandwidth" and "masking". In our method of calculating loudness given above, we neglected these properties. However, Zwicker [1957, 1965, 1991] has dealt extensively with these issues and has formulated a more accurate method of loudness calculation for complex sounds.

2.2.2 Performance Dynamics Related to Actual Sound Level Produced

Musicians are capable of producing a variety of amplitudes on acoustic instruments by performing according to "dynamic markings". The commonly used dynamic markings are from soft to loud (left to right) as follows:

... ppp pp p mp mf f ff fff ...

We might be tempted to assume that each of these markings corresponds precisely to either some physical intensity (given in Watts/m^2 at some reference distance from the musical instrument) or to some subjective loudness value. This might be true if musicians were trained to adhere to some objective standard of intensity or loudness production. A somewhat safer assumption would be to assume that each dynamic marking corresponds to an Intensity Level which is always a fixed number of decibels greater than that of the preceding, lower dynamic marking (e.g., **ff** would be 5 dB larger than **f**, **p** would be 5 dB greater than **pp**, etc.). However, actual measurements indicate that neither of these assumptions hold up.

In the early 1960's Melville Clark and David Luce made a comprehensive study of the dynamics produced by musicians performing various orchestral instruments [Clark and Luce, 1965]. Each musician was instructed to play a scale over the range of his instrument and play it as much as possible at the same dynamic marking; three dynamics were used, **pp**, **mf**, and **ff**. Recordings were made in an anechoic chamber (a chamber completely free of echoes or reverberation) with the distance to the microphone calibrated at 10 meters. Tables and graphs of intensity level vs. pitch (frequency) were made. Also, the graphs were fitted with smooth, interpolating polynomials in order to simplify interpretation of the data. The results for four instrumental families are shown in Figure 2.2.

Here are some of Luce and Clark's observations:

> "It is noted . . . that the dynamic range of the woodwind instruments is decidedly restricted compared with the dynamic ranges of other instruments. The intensities of the woodwinds usually fall between 50 and 60 dB for any dynamic marking. The dynamic range of the strings is somewhat greater (about 7 dB) than that of the woodwinds, although the average intensity of all string instruments is not greatly less (1.8 dB) than that for the woodwinds. The average intensity of the brass instruments is considerably greater (11.2 dB) than that of the woodwinds. The dynamic range of the string instruments is about the same as (0.3 dB greater) than that of the brasses. The average intensity of a scale played pianissimo on a nonhorn brass instrument is very approximately as loud as the average intensity of scales played fortissimo on strings or woodwinds."

> "We notice that the intensities of the French horn and the flute increase drastically with note number . . . The intensities of the violin and the viola and the double reeds are rather constant over their corresponding scales . . . The cello and the double bass are the only instruments for which the intensity consistently decreases with increasing note number."

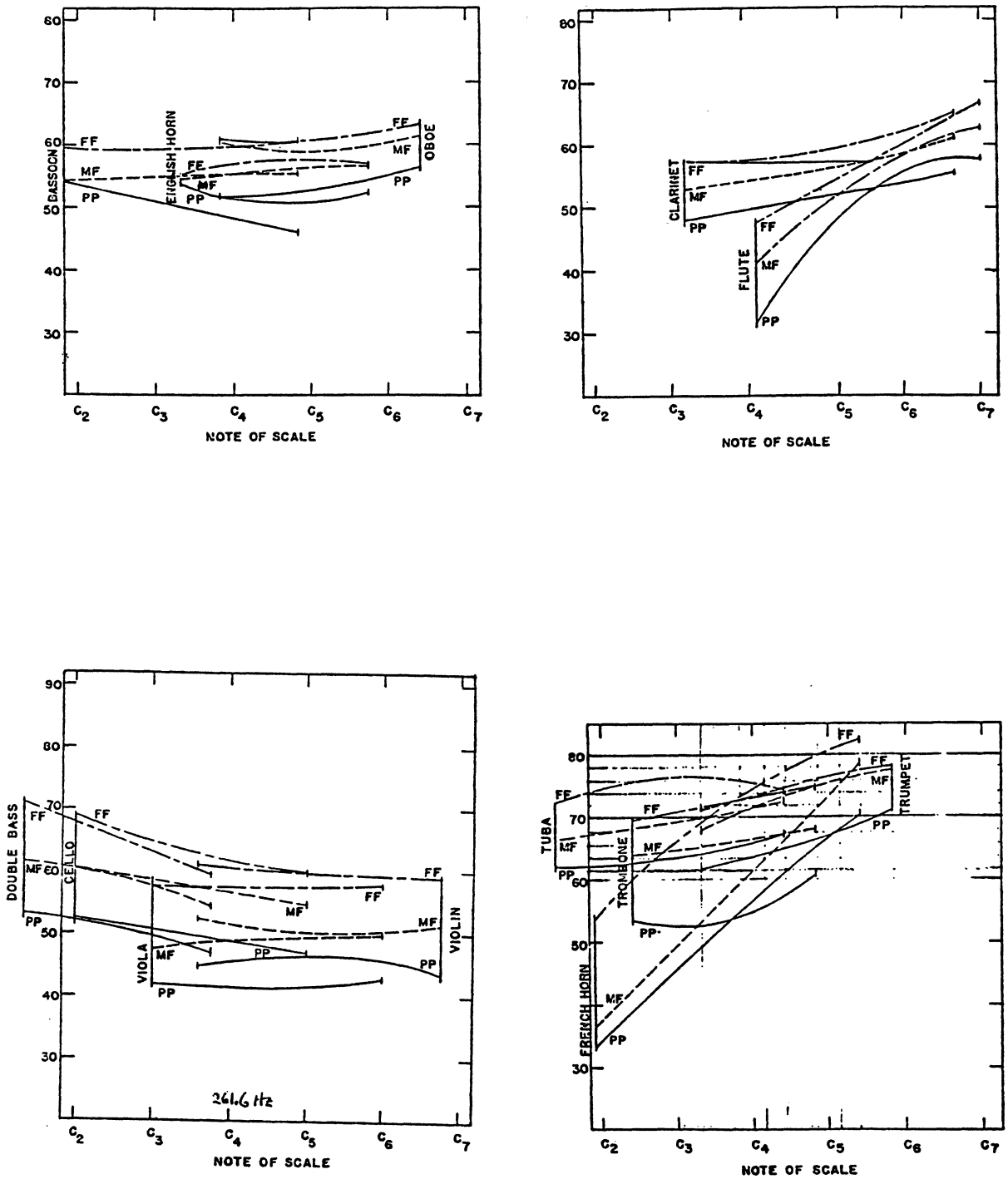


Figure 2.2 Intensity Levels in Decibels vs. Dynamic Markings and Performed Pitch for Various Musical Instruments (from Clark, Luce, JAES, 13, 151 (1965)).

The degree of consistency in performing a given dynamic was estimated by Clark and Luce to be the deviation between the actual intensity and that predicted by the smooth interpolating curve for any note, and these results were averaged over the entire scale for each dynamic of each instrument. These deviations (a measure of the randomness of intensity level production in dB) were rather consistently about 2.5 to 3.5 dB for all instruments except the non-horn brasses, which were consistent within 1 to 1.5 dB. The authors comment that "It would seem that the deviations of about 1 dB for the nonhorn brass instruments are so small that we must doubt if the player regulates the intensity of his scales aurally. We speculate that he may regulate the intensity by lip stress and/or internal air pressure."

The number of distinctly different dynamics (which Clark and Luce call quanta) available for a given instrument can be estimated by dividing its average dynamic range (in dB) by its average dB deviation. The result is between 4 and 6 quanta for the strings, between 2 and 5 quanta for the woodwinds and the French horn, and between 10 and 11 quanta for the brasses. The average for all instruments is about 5 quanta, a fortuitus result, since this is the number of dynamic markings which traditionally occur between **pp** and **ff**. However, the wide average dynamic range of the nonhorn brasses (about 12 dB) together with their relatively small deviations demonstrates that much smaller gradations of dynamics would be possible with these instruments. Note, however, that this does not take into account listener ability for discrimination of dynamics..

Another related paper on dynamics was contributed by Blake Patterson [1974]. Patterson noted that certain instruments favored by baroque composers, such as the recorder and the harpsichord, were later supplanted by instruments having greater dynamic ranges, the flute and the piano. This gave them an advantage in orchestral music, where the trend during the 16th to 19th centuries was towards enhancing the range of sound power of music in order to accomodate larger auditoria and audiences. After reviewing the work of Clark and Luce, he comments that while "good" (student) musicians may have narrow dynamic ranges, highly dedicated musicians can practice to achieve much greater ranges. (According to his results, several instrumentalists have been able to achieve ranges of 40 dB, more than three times Clark and Luce's average figure.)

References on Pitch, Tuning, and Loudness

1. Backus, John, "Intervals, Scales, Tuning, and Temperament", Chapter 8 in Acoustical Foundations of Music, Norton, 2nd ed. (1977).

2. Backus, John, "Frequency and Pitch", Chapter 7 in ibid. (1977).

3. Backus, John, "The Ear: Intensity and Loudness Levels", Chapter 5 in ibid. (1977).

4. Clark, Melville E., and Luce, David A., "Intensities of Orchestral Instrument Scales Played at Prescribed Dynamic Markings", J. Audio Engr. Soc., Vol. 13, pp.151- (1965).

5. Ellis, Alexander, "The History of Musical Pitch in Europe" in H.C.F. Helmholtz: Sensations of Tone, Dover, pp.494-513 (1885).

6. Flanagan, J.L. and Saslow, and Saslow, M.G., "Pitch Discrimination for Synthetic Vowels", J. Acoust. Soc. Am., Vol. 30, pp. 435-442 (1958).

7. Fletcher, Harvey, and Munson, W.A., "Loudness, Its Definition, Measurement, and Calculation", Acoust. Soc. Am., Vol. 5, pp.82-108 (1933).

8. Helmholtz, Hermann L. F., **Sensations of Tone as a Physiological Basis for the Theory of Music,** Dover Publications (1862, 1877, reprinted 1954).

9. Henning, G.B., and Grosberg, S., "Effect of Harmonic Components on Frequency Discrimination", J. Acoust. Soc. Am., Vol. 44, pp. 1386-1389 (1968).

10. Hunt, Frederick V., Origins in Acoustics, Yale University Press (1978).

11. Kinsler, Lawrence E.; Frey, Austin R.; Coppens, Alan B.; and Sanders, James V., Fundamentals of Acoustics, 3rd ed., pp.262-274 (1982).

12. Klatt, D.H., "Discrimination of Fundamental Frequency Contours in Synthetic Speech: Implications for Models of Pitch Perception", J. Acoust. Soc. Am., Vol. 53, pp. 8-16 (1973).

13. Moore, B.C.J., "Relation between the critical bandwidth and the frequency-difference limen", **J. Acoust. Soc. Am.,** Vol. 55, p. 359 (1974).

14. Patterson, Blake, "Musical Dynamics", Scientific American, pp.78-95 (Nov., 1974).

15. Shower, E.G., and Biddulph, R., "Differential Pitch Sensitivity of the Ear", J. Acoust. Soc. Am., Vol. 3, pp. 275-277 (1931).

16. Stevens, S.S, "The Relation of Pitch to Intensity", J. Acoust. Soc. Am., Vol. 6, pp. 150-154 (1935).

17. Stevens, S.S., "Calculation of the Loudness of Complex Noise", J. Acoust. Soc. Am., Vol. 28,

pp.807-832 (1956).

18. Wood, Alexander, The Physics of Music, Methuen, pp. 47-49 (1944).

19. Zwicker, E.; Flottorp, G.; and Stevens, S.S., "Critical Band Width in Loudness Summation", J. Acoust. Soc. Am., Vol.29, pp.548-557 (1957).

20. Zwicker, Eberhard and Scharf, Bertram: "A Model of Loudness Summation", Psych. Review, Vol. 72, pp.3-26 (1965).

19. Zwicker, Eberhard and Zwicker, U. Tilmann, "Audio Engineering and Psychoacoustics: Matching Signals to the Final REceiver, the Human Auditory System", J. Audio Engr. Soc., Vol. 39, No. 3, pp. 115 - 126 (1991).

2.3 Timing: Beats, Duration, Tempo, Rubato

A fundamental concept for measuring time in music is the "beat". This somewhat vague concept (from an objective point of view) has its roots in primitive music and can be correlated with psycho-physical functions of the human apparatus. For performance, it reasonably clear what is meant by "beats per minute" and the "duration of a beat". However, for a listener (or for an acoustic analysis machine) it is not always clear "where the beats are".

Beats are arbitrary points in time used as references for the performance of "notes". In traditional music, it usually quite clear from the score where the beats lie. In some modern music, the position of the beats may not be so obvious. For one thing, a contemporary composer may choose to overlay two beat patterns which have a complex relationship to one another. Under such conditions, for the listener, the position of the beats may seem to disappear and reappear, depending on the listener's ability to focus or "channel" on one particular sound layer. In delayed-performance electronic music there is no absolute necessity for using the beat concept; nevertheless, beats are extremely useful tools for organizing the temporal aspects of music.

Within a beat we have "notes", which are characterized by their pitches and their durations. In traditional notation the conventions for indicating duration are quite consistent and lend themselves to mathematical translation rather nicely. So, it is rather easy to develop computer programs to input musical notation (either graphically or using sequences of alpha-numeric characters) and have these translated into actual timings of beats and notes. This can be extremely useful for coded-performance synthesizers (e.g., MIDI) or for the synchronizaton of synthetic music and video, since the latter usually relies on timings in seconds.

If we start with a quarter note, its duration is halved for each addition of a flag. The half note and whole note double and twice-double, repectively, the duration of the quarter note. So, as in musical pitch, x2 and x_.5 operations play a leading role. There are two ways to alter durations by other than powers of two: First, a "dot" can be used after a note to multiply its duration by 3/2. In general, n dots can be used to multiply a duration by the factor

$$(2 - 2^{-n}).$$

A second method is to use "triplets", "quintuplets", etc; triplets simply divide a note into 3 equal durations whereas quintuplets divide it into 5 parts. {The notation is something else, however; the number of flags used is according to the next lower power-of-two duration.} An extension of this concept is to have notes which occur at the rate of, for example, "four in the time of seven". I.e., four notes occur in the time of seven beats (or sub-beats) previously defined. In reference to the duration of the seven, the durations of the four would be 7/4th of the seven durations. We see that things can get quite complicated on the level of specifying individual note durations.

Bars are used to delimit groups of beats, and the number of beats per bar as well as the length of note used to represent the beat is indicated by the "time signature". Therefore, "4/4" indicates that there are 4 beats in a bar and that a 1/4th note or quarter note takes up one beat. "6/8" indicates that there are 6 beats in a bar and that a 1/8th note or eighth note takes up one beat. (However, it is traditional to perform this time signature as two beats which are subdivided into triplets.) For modern, free tempo music, time signatures are not very useful, as they would have to be changed too often throughout the piece. For coded-performance music, bars are not really necessary, although they can be helpful for error checking

purposes.

Tempo is usually indicated by terms like **MODERATO**. Although such terms are subject to a wide range of interpretation, they can be translated into concrete values of tempo in beats-per-minute. For our purposes tempo will be more conveniently measured in beats-per-second. To understand this with more precision let us make some definitions:

1) Let B(t) be a function of time which gives the beat number starting with B=0 at t=0. Whenever B(t) equals an integer, a "beat" occurs.

2) Let Temp(t) be the instanteous tempo in beats-per-second. It follows that

$$\text{Temp}(t) = dB/dt \qquad [2.3.1a]$$

$$\text{and } B(t) = \int_0^t \text{Temp}(t)dt \qquad [2.3.1b]$$

2.3.1 Calculation of Beat Timings Under Conditions of Changing Tempo

Much music expects or specifically indicates that its tempo is to change (modulate) over a period of time. Accelerando and ritard) are terms which designate smooth increase and decrease in tempo over time. All sorts of functions can be used to accomplish this type of change. For example, we can make a linear change of tempo from Temp_0 to Temp_1 over the time interval $\{0,t_1\}$ as follows:

$$\text{Temp}(t) = \Delta\text{Temp}\ t/t_1 + \text{Temp}_0. \qquad [2.3.2a]$$

where $\Delta\text{Temp} = \text{Temp}_1 - \text{Temp}_0$ is the tempo change.

We can integrate this expression and derive the value of the beat function as it changes with time:

$$B(t) = \Delta\text{Temp}\ t^2/(2\ t_1) + \text{Temp}_0\ t \qquad [2.3.2b]$$

Plots of a linearly changing tempo function and corresponding beat function are shown in Figure 2.3. A "beat" occurs when the beat function takes on an integer value, i.e., 0, 1, 2, 3, etc. The beat function formula can be solved to determine the exact times the beats will occur:

$$t = \{t_1\ \text{Temp}_0/\Delta\text{Temp}\}\ \{\text{sqrt}[1 + 2\ \Delta\text{Temp}\ B(t)/(t_1\ \text{Temp}_0^2)] - 1\} \qquad [2.3.3]$$

In most cases we will know the number of beats which elapse during the tempo change; since the elapsed time is t_1, the corresponding beat value is $B(t_1)$. Substituting t_1 for t in Equation 2.3.2b yields

$$B_1 = B(t_1) = t_1\ (\text{Temp}_0 + \text{Temp}_1)/2 \qquad [2.3.4a]$$

$$\text{or } t_1 = B_1/\text{Temp}_{ave} \qquad [2.3.4b]$$

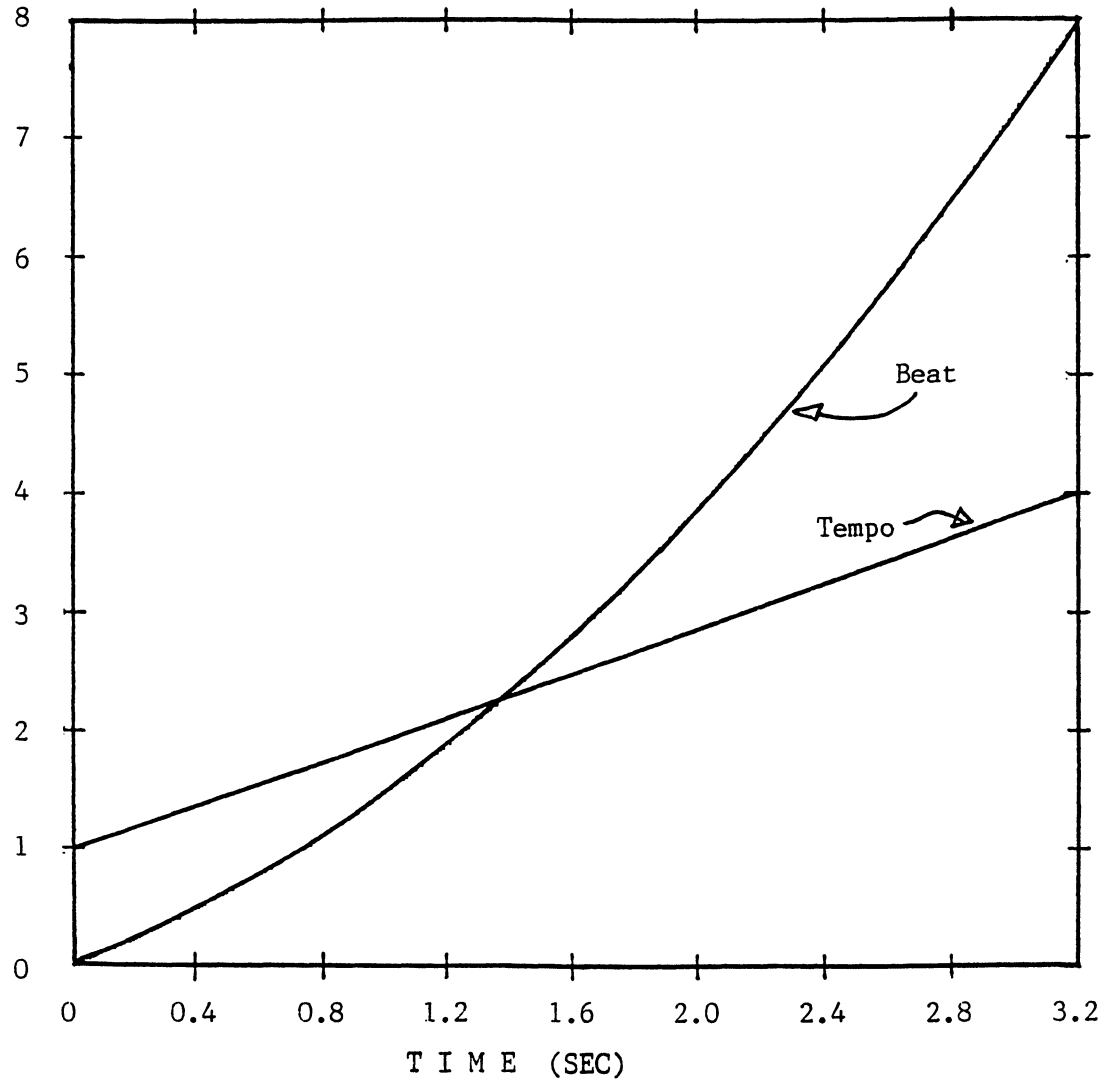


Figure 2.3 Tempo and corresponding Beat-vs.-Time functions for an accelerando from 1 beat/sec to 2 beat/sec over 4 beats.

This value of elapsed time, which depends on the number of elasped beats, can be calculated prior to using Equation 2.3.3 to calculate the beat times.

As an example of a beat timing calculation for a specific amount of accelerando, let's consider a situation where the tempo increases linearly over 5 notes from 1 bps for beat 0 to 2 bps for beat 4. Since the average tempo is 1.5 bps, we can use Equation 2.3.4b to obtain the elapsed time t1 = 2.667. We can then obtain an equation for time t in terms of beat B from Equation 2.3.3, namely,

$$t = 2.667 \{\text{sqrt}[1+.75B] - 1\}.$$

For this case the table below gives the beat times and beat durations as well as the instantaneous tempo using Equation 2.3.2a:

B	0	1	2	3	4
t	0	.861	1.550	2.141	2.667
dur	.861	.689	.591	.526	.500
Temp	1	1.323	1.581	1.802	2.000

Another possibility is for the tempo change to be exponential:

$$\text{Temp}(t) = \text{Temp}_0 (\text{Temp}_1/\text{Temp}_0)^{(t/t_1)}$$

$$= \text{Temp}_0 e^{\alpha t} \qquad [2.3.5a]$$

which integrates to give

$$B(t) = (\text{Temp}_0/\alpha) \{e^{\alpha t} - 1\} \qquad [2.3.5b]$$

From Equation 2.3.5a, we can calculate the final tempo as

$$\text{Temp}_1 = \text{Temp}(t_1) = \text{Temp}_0 e^{\alpha t_1} . \qquad [2.3.6a]$$

Solving this equation for α gives

$$\alpha = \{1/t_1\} \ln(\text{Temp}_1/\text{Temp}_0) . \qquad [2.3.6b]$$

By substituting Equation 2.3.6b into Equation 2.3.5b and setting $B = B_1$, the elapsed time may be calculated as

$$t_1 = (B_1 /\Delta\text{Temp}) \ln(\text{Temp}_1/\text{Temp}_0) , \qquad [2.3.7]$$

which when substituted into Equation 2.3.6b yields

$$\alpha = \Delta\text{Temp}/B_1 . \qquad [2.3.6c]$$

Finally, we can solve Equation 2.3.5b to find the time for each beat:

$$t = (B_1/\Delta Temp)\ \ln[1 + (\Delta Temp/Temp_0)\ (B/B_1)]\ . \qquad [2.3.8]$$

Heretofore we have discussed tempos which are defined to change with respect to time. We can also consider tempos to be defined at various points in a piece, i.e., in terms of the **beats** themselves. We may think of the beats as a linear grid on a rubber sheet which may be stretched (or compressed) in a nonuniform fashion depending on a **tempo warping function** [Jaffe, 1985], i. e.,

$$Temp(t) = f(B(t)) \qquad [2.3.9a]$$

Another linear grid superimposed on the warped sheet would determine the time which occurs at each beat. Mathematically, we can solve for time t for specific warping functions. Consider linear and exponential warping:

$$Temp(t) = \quad dB/dt = Temp_0 + \Delta Temp\ B/B_1 \qquad [2.3.9b]$$

$$Temp(t) = \quad dB/dt = Temp_0\ e^{(B/B_1)*\ln(Temp_f/Temp_0)} \qquad [2.3.9c]$$

The reader can solve these equations and determine the beat timings and durations for specific cases.

Four different formulations were given above for performing accelerandos (or ritards). They will yield somewhat different results under most circumstances. However, only if a tempo change takes place over a long period of time and is a quite drastic change will there be noticable differences between the various methods.

2.4 Methods of Coding Music

The idea of coding music comes up whenever one considers the use of computers to generate music or to print musical scores, particularly for sending music data in coded form over long distances or for control of synthesizers. There is a question whether the code should be easily readable by humans or whether it should be couched in a more compact form that only machines can understand. Both types of codes have been used, and sometimes it is possible to easily transpose between the two types.

In synthesis we are concerned with the control of the parameters of musical sound. A complete code allows detailed aspects, such as timbre or glissando, to be specified in great detail. In order to demonstrate some of the challenges of music coding we will look at some methods which have been or are still being used.

2.4.1 Common Music Notation

Common music notation (CMN) can used as a code for music synthesis. However, it has two problems: 1) CMN is not complete. I.e., it does not provide enough symbols to specify all of the acoustic parameter changes which occur in a sophisticated performance. 2) It is very difficult to automatically convert musical sound into CMN, even for approximate renditions. However, CMN is very effective for human performance. People are very efficient 2D graphic readers and are capable of intelligently "filling in the gaps". Because CMN has been the standard for a long time, it is represented by a massive literature and millions of individuals who immediately respond to it to produce stunning performances.

Coding of music for the purpose of printing scores involves a somewhat different set of problems than that required for music synthesis. In score copying or printing one must must include many special symbols which allow a score to be interpreted by performers in a sophisticated manner. For example, slurs and stems are not trivial to form graphically. Codes must be provided for all of these symbols, and these must be translated into actual graphics. A complicating factor is that symbol formation is dependent on the context in which the symbols are embedded. E.g., the length and shape of the slur symbol is dependent on the number of notes to be slurred.

During the 1980s microcomputer programs were developed which allow an internal representation of graphic musical scores and a "playback" of a score via an internal synthesizer or via MIDI. Usually the playback is used to check the accuracy of the score. Sometimes the score may be used to document a MIDI performance. The principal limitations of these systems are that sophisticated use of symbols in the graphic score do not necessarily map into sophisticated parameter manipulations in the synthesis, and vice versa. For example, scoring programs usually do not attempt to graphically represent the effect of the pitch bend wheel within a score.

2.4.2 Software Synthesis: "Music X"

Early in the history of computer music a method of coding notes primarily with numbers was devised. More importantly, a program was developed which allowed coding of arbitrary algorithms to produce a wide variety of sounds. "Music X" refers to a genre of music synthesis computer program which was pioneered by Max Mathews at Bell Laboratories during 1957-63. From Mathew's early prototypes, several offspring have been developed such as Music 4, Music 10, Music 11, Music 7, and Music 360 (written in macro assembler); Music 4BF and Music 5 (written in Fortran); and C Music, C Sound, and M4C (written in C). All of these programs share three characteristics: 1) First, instrument timbres are

defined as interconnections of basic synthesizer blocks called "unit generators", analogous to the way that analog synthesizers modules may be patched together. 2) The score for a piece consists of a series of notes (or "events") each of which gives the instrument No. (or name), a start time, duration, and a set of parameter values (called P1, P2, ...) which are valid for that note. 3) Sound is generated in two steps: computation of "samples" (usually in non-real time) followed by real time digital-to-analog conversion of the samples, although the two steps could be accomplished concurrently if the computer were sufficiently fast.

Instruments (or voices) are generally defined using a language which is close general-purpose. Thus, an immense variety of timbres can be programmed by composers or instrument designers. The code for an instrument definition can be given in three parts. The first routine defines acoustic parameters to be used throughout the piece. The second routine converts the score parameters Pk into parameters used for sample calculation. The third part computes one sample of the output signal for one voice. Here is an example of a higher level (C language) definition of a simple instrument which generates a sine wave with amplitude envelope; it is programmed by 5 parameters -- duration, frequency, amplitude, attack time, and decay time.

```
Sine Wave Instrument Definition for Music X

#define MAXV 3
#define TABSIZE 512
float P[100], Sine[TABSIZE], freq[MAXV], amp[MAXV],
frpitch(), Envel(), Osc();
ENVSTATE *Set_env(), *envstate[MAXV];
OSCSTATE *Set_osc(), *oscstate[MAXV];
Tone_init()
{ int k;
  for (k=0;k<TABSIZE;k++) Sine[k] = sin(k*2.*PI/TABSIZE);
}
Tone_set(int N)  /* N is the instrument number */
{ float dur,,tattack,tdecay,tss;
  dur = P[3]; freq[N] = frpitch(P[4]); amp[N] = P[5]
  tattack = P[6]; tdecay = P[7]; tss = dur - tattack - tdecay;
  env_state[N] = Set_env(tattack,tss,tdecay)
  osc_state[N] = Set_osc(Sine,phase[N]);
}
Tone_samp(int N)
{
  output(amp[N]*Envel(env_state[N])*Osc(freq[N],osc_state[N]));
}
```

Intializations are accomplished by Tone_init and Tone_set, while the actual sample computation is accomplished by a nested computation inside the routine Tone_samp. Prior to invoking Tone_set, P fields 3 through 7 of a score statement (see below) are read into the P[] array. Tone_set's purpose is for each note to translate these "user-friendly parameters" into parameters convenient for sample computation by Tone_samp (see the line beginning with "output"). These are initial values of state variables using

array value N, where N is the instrument number being played. Various functions are called (e.g., Setenv() and Osc()), and the instrument designer must understand how these work. The state variables are for use during sample computation and are updated during each sample computation. The music program takes care of scheduling of events and combining of various instrument voices to form the final output stream.

Sample computation algorithms are actually very analogous to analog synthesizer patches, and it is useful to represent them as flow diagrams. A flow diagram for the envelope-controlled sine wave instrument is shown in Figure 2.4.

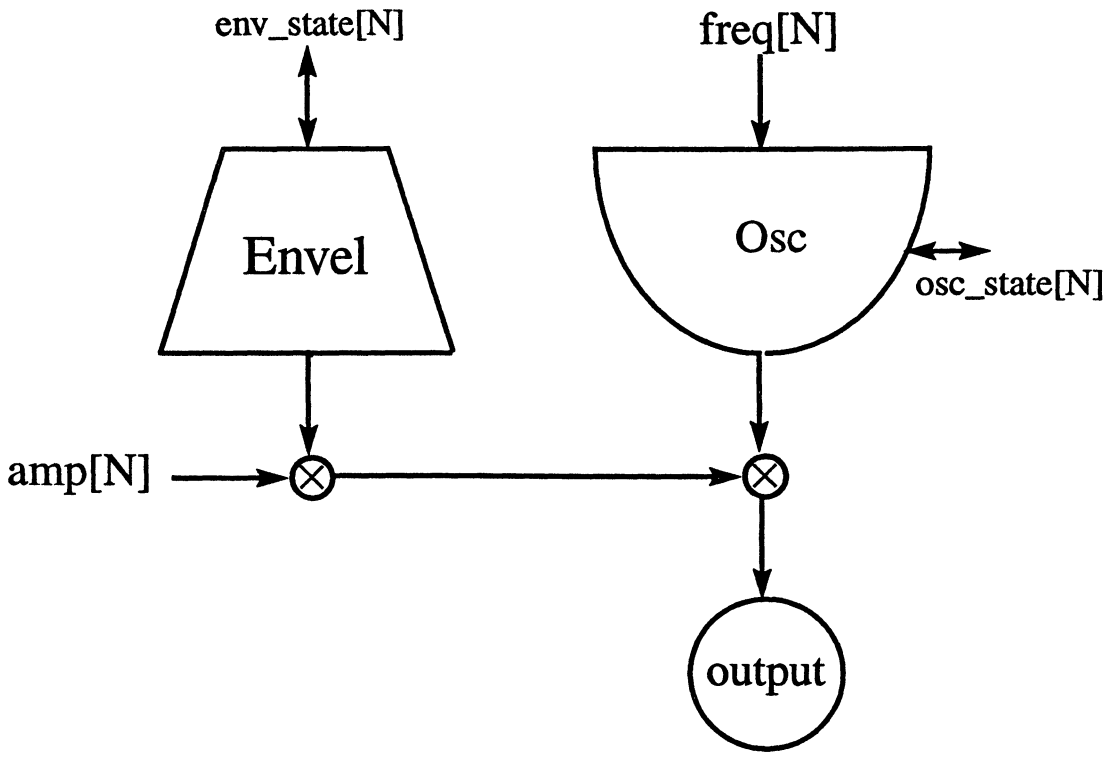


Figure 2.4. Flow Diagram for Envelope-Controlled Sine Wave Instrument N refers to the instrument No. env_state and osc_state refer to registers which hold values for table lookups and increments.

In contrast to the instrument definition, which requires some knowledge of the acoustics of patch design, a score to program notes played by this instrument is very simple to understand.

Example Score for Music X

C	P1	P2	P3	P4	P5	P6	P7
C	insno	stime	dur	pitch	amp	attack	decay
I	1	0	1	4.09	1000	.03	.35
I	2	.5	1.5	4.051	1200	.02	.4
I	3	.75	1.2	4.00	1500	.01	.5
I	1	1	.5	4.033	1000	.02	.2
E							

Each "I card" (instrument statement) begines with the op code I and gives the instrument number, start time, duration, pitch (in octave.pitch notation), amplitude, attack time, and decay time of a note to be played. Note that whereas '4.09' refers to an equal-tempered A4 (the 9th note of the fourth octave), 4.051 represents an F4 which is 10 cents sharp. All times are given in seconds. There is no limitation on pitch or time accuracy with this system. Op codes C and E are used for comments and score termination, respectively. Except for the opcodes, scores are simply lists of numbers, admittedly a tedious but, still, a flexible method of representing music. One convenience of non-real-time is that the statements need not be given in order of start time. Music X automatically sorts the statements according to time.

In summary, the Music X approach is very versatile, since instrument designs are a matter of writing code in a high level language. Function calls are used to simplify the coding, but the user must understand how these functions work. The scoring system is simple to understand but tedious to carry out without the aid of a computer.

2.4.3 A Method for Alphanumeric Representation of Music Notation: Notepro

Several methods for computer entry of conventional music notation have been devised since the mid 1960's. Two examples are DARMS [Erickson, 1975] and SCORE [Smith, 1972]. Notepro is a language for encoding conventional music notation using standard ASCII characters which has used at the UIUC Computer Music Project for several years. Each of the letters, **A, B, C, D, E,** and **F** represent basic quarter notes in the middle octave (between C4 and B4). **R** represents a quarter note rest. Tuning defaults to standard equal-tempered but can be varied. Pitch and duration modifiers are placed to the **right** of each note symbol to achieve the correct octaves and/or accidentals and fractions (or multiples) of the quarter note. These modifiers may be combined in any mixture after a note symbol to achieve the desired effect.

Pitch modifiers:

(sharp) raises the note frequency by $\times 2^{1/12}$
b (flat) lowers the note frequency by $\times 2^{-1/12}$
^ (up-octave) raises the frequency by $\times 2$
v (down-octave) lowers the frequency by $\times .5$
q (quarter tone sharp) raises the frequency by $\times 2^{1/24}$ (+50 cents)

+ (microtone sharp) raises the frequency by $\times 2^{1/120}$ (+10 cents)
- (microtone flat) lowers the frequency by $\times 2^{-1/120}$ (-10 cents)

Duration modifiers:

/ (flag) lower the note duration by $\times .5$
o (half) raise the note duration by $\times 2$
. (dot) raise the duration by $\times 2 - (1/2)^n$, where n is the number of cumulative dots.
n (n-tuplet) lower the duration by $\times 2^{\text{int}[\log 2(n)]}/n$.
Example: n = 7 results in $\times 4/7$ (septuplets)
X:Y ("X in the time of Y") modify the duration by $\times (X/Y)$

Duration and pitch holds:

[and { brackets are used to initiate pitch and duration modifier "holds" over a group of notes. The modifiers are placed immediately to the **right** of the hold brackets, and they affect all subsequent notes until the respective closing bracket is encountered. Brackets may be nested, and their effect is cumulative. A right bracket will only turn off the effect of the previous left bracket of the same type. The primary purpose of the pitch hold feature is to avoid having to place up-octave (^) or down-octave (v) symbols next to many notes in succession which happen to be in a high or low register. For durations, the use of the duration hold saves many characters for passages involving groups of short or long notes.

Ties:

The symbols (and) are used to encompass a group of notes which are to be tied together at the same pitch. The first note within the group sets the pitch for the tie. An example would be

(A^#+o A///. A////3)

Dynamics:

A dynamic gives the amplitude for each note occuring after the dynamic until another, different dynamic occurs in the note stream. Some dynamic symbols available are **p** (piano), **mp** (mezzo piano), **mf** (mezzo forte), and **f** (forte). The dynamics in order from softest to loudest are

... **pppp ppp pp p mp mf f ff fff ffff** ...

where the amplitude spacing is 6 dB ($\times 2$).

Crescendo/decrescendo can be coded by use of < or >. A gradual dynamic change from the current dynamic value continues until the next dynamic is encountered.

Tempo:

A constant tempo is specified at any point by the symbol **T** followed by a number giving the tempo in quarter-notes per minute. The default tempo is 60 beats/min. An accelerando (or ritard) is initiated by the occurence of a T with no number following. The tempo change continues until another **T**n occurs, where n gives the target tempo.

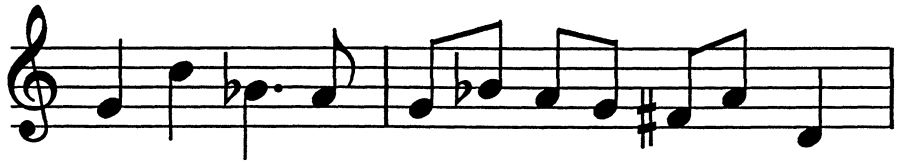
Voice or instrument designator:

The voice or instrument number or name playing the sequence of notes which follows is deliminated by single quotes (e.g., 'voice1'). The appearance of the instrument voice name resets all defaults including time, which is reset to zero, and tempo, which is reset to 60 beats/min.

Time-tempo Save/Reset:

It is important to be able to save time and tempo at various points in a note stream so that subsequently indicated voices can enter at these times. This saves calculating large rest values. The time and tempo can be saved in "register n" at any point by asserting tn, where n is an integer. Subsequent points that tn is mentioned result in times and tempos being reset to the time stored in register n. Each "register" can be used to save time only once in the score, but it can reset as many times as desired.

An example of Notepro coding is shown in Figure 2.5. Notepro has been implemented with the Placomp Synthesizer [Murray et al, 1978] and with Music 4C [Beauchamp, Code, and Chen, 1990] , and with Adagio of the CMU MIDI Toolkit [Dannenberg, 1986].



Notepro score:

```
G   D^   Bb.  {/ A  |  G  Bb   A   G   F#   A } D
```

Notepro output (Music X input):

C	insno	stime	dur	pitch	amplitude
I	1	0.0	1.0	8.07	1024
I	1	1.0	1.0	9.02	1024
I	1	2.0	1.5	8.10	1024
I	1	3.5	0.5	8.09	1024
I	1	4.0	0.5	8.07	1024
I	1	4.5	0.5	8.10	1024
I	1	5.0	0.5	8.09	1024
I	1	5.5	0.5	8.07	1024
I	1	6.0	0.5	8.06	1024
I	1	6.5	0.5	8.09	1024
I	1	7.0	1.0	8.02	1024
E					

Figure 2.5 Example of Common Music Notation translated into Notepro code which is in turn converted into a Music X numerical format.

2.4.4 A Real Time Code for Synthesizers: MIDI

The **Musical Instrument Digital Interface** standard was developed during 1983 by a consortium of synthesizer manufacturers. The standard encompasses the electronic interface as well as timing and musical information coding. We will only mention here that the connector is normally a 5 pin DIN, but only 3 pins are used (2 for twisted pair current loop, one for shield grounded to chassis at the MIDI OUT end), that the serial baud rate is 31,250 bits/sec, and that a 5 ma. current loop connection is used. Thus, the interface is cheap and quite impervious to noise. Unfortunately, it is not compatible with RS232.

MIDI code is designed in an attempt to use the lowest information rate to program the most functionality; i.e., those functions which must be changed most often (e.g., pitch change) are given the highest priority and take the smallest amount of code to initiate. Unlike the Music X score, there is no mention of time in the code. This is because (within the limits of the equipment) the occurrence of a command calls for immediate action. The original goal of MIDI was to allow synthesizers to talk to one another. When a key is depressed, pitch and velocity information must be immediately sent to another synthesizer; there is no time to figure out start times and durations.

Data consists of a series of 10 bit packets, whose first and last bits are for START and STOP, so only 8 bits (one byte) are actually significant. There are two types of packets, **STATUS** bytes and **DATA** bytes, where the job of the status bytes is to indicate what the data bytes which follow are used for.

Status and data bytes are distinguished by their leading bits, 1 and 0, respectively, reducing the significant information-bearing bits to 7. Status bytes are generally used to specify the meanings of the (one or more) data bytes which immediately follow. The 7 significant bits of the status byte are divided into two parts: 3 bit **action code** and 4 bits which usually indicate a **channel number**, but for the case of action code 111 (7) indicates a **system message**.

Channel number generally refers to one particular synthesizer of several which might be used together. At the time of this writing, most synthesizers are designed to produce several simultaneous notes but only one timbre, and thus are restricted to one channel only. However, there are a few "multi-timbre" synthesizers which can receive on two or more simultaneous channels (e.g., the Casio CZ-101 and the Yamaha TX81Z and FB01). A single MIDI stream is limited to sixteen channels. On the surface this means a limit to 16 different timbres. However, two ways to circumvent this problem are 1) the split keyboard method and 2) use of multiple MIDI streams in parallel.

The most important status bytes indicate **note-on** (status code 001) and **note-off** (status code 000). Either one is followed by one or more pairs of bytes which specify **pitch** and **velocity**. Valid pitch values are in the range 0 - 127 (7 bits) in half-step increments with a value of 60 assigned to C4 (middle C). Thus pitches (frequencies) from C-1 (8.2 Hz) to G9 (12544 Hz) can be programmed. Note that this system does not allow for microtones. Valid note-on velocity values (which may result from how hard a keyboardist strikes his keys) also range from 0 to 127, and these are generally mapped into amplitude. However, we know of no standard for this mapping. A special case is note-on velocity = 0: This can be used to turn a particular pitch off, as an alternative to using the note-off command. Since a single note-on command can be followed by any number of pitch/velocity byte pairs without intervening status bytes, this method results in less data for the desired result. The real usefulness of the note-off command is to allow a variation of the note release time using the note-off velocity parameter. However, most synthesizers do not include the note-off velocity feature.

Other status bytes are used for changes of other synthesis parameters. Status code 010 is used for **Polyphonic Key After-Touch**. A 7-bit pitch data byte is followed by a 7-bit value which corresponds to the pressure on an individual key. This is an expensive feature to implement. On the other hand, status code 101 is used for Channel **After-Touch**. Here a single 7-bit value gives the pressure (amplitude) associated with the entire keyboard or channel. Code 110 is for the **Pitch Bend Wheel Change** where two 7-bit bytes (14 bit value) are used to program a fine scale pitch change. However, the pitch bend applies to all notes currently playing on a channel. Only if different channels are used is it be possible to program microtonal chords, and a further complication is that the pitch-bend mapping is not standardized. Finally, status code 011, **Control Parameter Change**, introduces a series of sub-codes, depending on the value of the data byte which follows. One of the sub-codes occurs when this data byte's value is between 122 - 127. Each of these values causes the receiving synthesizer(s) to select a particular "mode", e.g, OMNI, MONO, or POLY.

A very important action code is number 100, which is used for selection of **preset timbre patches**. The 7-bit data byte which follows allows for 128 possibilies for a given channel. These presets may have been made at the factory or they may have been designed and implemented by the user. Remote timbre design using MIDI is possible, using a feature called **system exclusive** (see below), which depends on the individual synthesizer manufacturer.

All of the status byte codes mentioned above are associated with 4 status bits which give the channel number. An exception is status code 111, which is for system messages. Instead of a channel number, the remaining 4 bits are used to indicate the message type. Some of the message types are given below:

Message Number	Description
0	System exclusive
2	Song position pointer (system common)
3	Song select (system common)
6	Tune request (for tuning analog oscillators)
7	End system exclusive (system common)
8	Timing clock (real time)
10	Start (real time)
11	continue (real time)
12	Stop (real time)
14	Active sensing (real time)
15	System reset (real time)

We see that compared to Music X and Notepro, MIDI code is a highly complex, not altogether logical, lower-level code. MIDI has been criticized for its lack of two-way communication, its over-emphasis on pop music performance considerations, the difficulty of playing different timbres and microtones, and, particularly, its limited baud rate, which can be easily swamped by overabundant pitch bend information. Nevertheless, it is a standard around which millions of devices have been designed and sold. Its scope of applications has far exceeded that of keyboard synthesizers, for which it was originally intended. So it is necessary to understand its intricasies in order to realize its possibilities and limitations.

Since the MIDI standard was originally designed with inter-synthesizer communications in mind, the importance of computer control was not foremost in the designers' minds. The arbitrary emphasis on key pressure, channel pressure, and pitch bend wheel, rather than simply amplitude and pitch, seems strange

from the point of view of a composer using a computer to control a rack-mounted synthesizer. However, MIDI has opened up a new industry -- computer software for synthesizer control. Soon after MIDI was introduced, MIDI interfaces for various popular microcomputers became available. During 1985, software was developed to analyze and synthesize samples signals on a microcomputer, which could be downloaded oruploaded to sampling synthesizers via the MIDI channel; this was made possible by the system exclusive feature of MIDI.

While the original MIDI specification included no mention of formats for storage of MIDI data in computers (as files) or for dumping samples (an important feature for sampling synthesizers), the MIDI Manufacturer's Association has recently published standards for these formats, meaning that the output of one manufacturer's program can be routed to the input of another's. Several types of products have become standrad: **sequencer-recorders, music printers, interactive composers, patch editor/librarians,** and combinations of these. In 1988, sequencers and sequencer programs/interfaces became available which could handle several synchronized MIDI signals, thus augmenting the number of simultaneous different timbres to well beyond 16, the limit for one signal. So far the MIDI standard has been sufficient to allow a continual expansion of music utility. It will be interesting to see how long this trend can continue before it will be necessary to modify or replace the standard.

References on Music X, Notepro, and MIDI

1. Beauchamp, James; Code, David; Chen, Kwok-Ping, "Notepro 3.0: A Music Transcription Tool for Computer Music Programs", Computer Music Project, School of Music, University of Illinois at Urbana-Champaign, Urbana, IL.

2. Dannenberg, Roger B., "The CMU MIDI Toolkit", Carnegie Mellon University, Pittsburgh, PA 15213 (1986).

3. Erickson, Raymond, "The DARMS Project: A Status Report", *Computers and the Humanities,* Vol. 9, pp. 291-298 (1975).

4. Howe, Hubert, Electronic Music Synthesis, Norton Press Chapters 7 & 8 (1975).

5. Jaffe, David, "Ensemble Timing in Computer Music", *Computer Music J.*, Vol. 9, No. 4, pp. 38-48 (1985).

6. Loy, Gareth, "Musicians Make a Standard: The MIDI Phenomenon", *Computer Music J.*, Vol. 9, No. 4, pp. 8-26 (1985).

7. Mathews, M.V., "An Acoustical Compiler for Music and Psychological Stimuli", *The Bell System Technical J.*, Vol. 40, pp. 677-694 (1961).

8. Mathews, M.V., "The Digital Computer as a Musical Instrument", *Science,* Vol. 142, pp. 553-557 (Nov., 1963).

9. Mathews, M.V., *The Technology of Computer Music*, M.I.T. Press (1969).

10. Moore, F. Richard, "The Dysfunctions of MIDI", *Computer Music J.*, Vol. 12, No. 1, pp. 19-28 (1988).

11. Murray, David J.; Beauchamp, James; Loitz, Gary, "Using the PLATO/TI980A Music System: The PLACOMP Language", *Proc. of the 1978 Int. Computer Music Conf.*, Northwestern Univ. Press, pp. 151-166 (1979).

12. Moog, Bob, "MIDI: Musical Instrument Digital Interface", *J. Audio Engr. Soc.*, Vol. 34, No. 5, pp. 394-404 (1986).

13. Rona, Jeffrey, "A Recording Engineer's Guide to MIDI", *Recording-engineer/Producer*, pp. 125-129 (December, 1983).

14. Smith, Leland, "SCORE--A Musician's Approach to Computer Music", J. *Audio Engr. Soc.*, Vol. 20, pp. 7 - 14 (1972).

Timbre, Waveform Analysis, Fixed Waveform Synthesis

CONTENTS

TIMBRE AND FIXED WAVEFORM ANALYSIS AND SYNTHESIS

3.0 Introduction

Timbre is a term which describes our perception of sound quality and is generally recognized to be closely related to the physical *sound spectrum*. It is a more complex and ambiguous attribute than pitch, loudness, and duration. The American Standards Association defines timbre as

> "that attribute of auditory sensation in terms of which a listener can judge that two sounds similarly presented and having the same loudness and pitch are dissimilar".

In other words, after a sound's pitch, loudness, and duration are determined, what is left is timbre, a "multi-dimensional grab bag" of many sub-perceptions. Timbre depends on several factors such as the duration of attack and release, rate of vibrato, ratios between partial frequencies, and amount of noise content, but probably the most important factor is the sound spectrum.

One general result of timbre perception is generally credited to George Ohm who in 1843 conjectured that the timbre of a steady tone depends solely on the amplitudes of its constituent sine waves (the amplitude spectrum) and not at all on their phases. This conjecture predicted that waveforms which differ in shape, but have the same amplitude spectrum, sound the same . Hermann Helmholtz confirmed "Ohm's acoustic law" to his own satisfaction by making careful listening tests in 1863. Nevertheless, a suspicion remained that phases had some effect on sound quality. This controversial issue was clarified considerably through an intensive study made by Plomp and Steeneken [1969]. They found that, although with careful listening it is possible to detect phase differences, the effect of phases is quite small for periodic tones with spectra representative of speech and music. Note that this result pertains only to *static* phase values. Phases which vary with time can be translated into equivalent frequency modulation; thus, we should expect that phase modulations of sufficient magnitude will have a substantial effect on a sound's timbre.

In recent years psychoacoustics research has been applied to determine the connection between perception of timbral attributes and certain physical parameters. One goal has been to determine, by means of listening tests, a "timbre space" containing a large set of known sounds which has easily identifiable timbral "dimensions" (Grey, 1975; Wessel, 1979). These dimensions, which are interpreted in terms of certain perceptual attributes, have been found to correspond to certain physical parameters or combinations thereof. Typically, in a two-dimensional solution, one dimension is interpreted as "brightness", which corresponds to the *spectral centroid*; the other dimension is interpreted as " tonal onset", which corresponds to *attack time*. Taking this idea a step further, one could construct a synthesizer with "knobs", corresponding to perceptual dimensions, used to control the positions of sounds in the timbre space, and the manipulation of several knobs could be used to produce interesting "timbral trajectories". However, it is still the case that there still is no adequate theory of the fundamental dimensions of timbre (analogous to primary colors in art). Composers and musicians usually find it easier to select amongst a palette of pre-designed timbres than to construct timbres from scratch.

One approach to the study of timbre is to analyze the spectra of sounds of acoustical musical instruments as played by expert practitioners. From analysis of a large group of sounds from the same instrument, it is possible to make generalizations about the spectra and develop a synthesis model which is controlled by a group of parameters. The parameters can then be mapped to perceptual parameters. Another possibility

is to perform "reverse engineering" of successful synthesizer patches, which if systemmatically employed could lead to some results in timbre theory. Still another approach is for a musician, composer, or synthesizer patch designer to put himself in a tight feedback loop with a sound synthesis instrument and gradually adjust parameters until he achieves sound qualities close to those which he desires, thus building up an intuitive connection between timbre (the perception) and the control of the sound spectrum. This is how most factory synthesizer patches are developed.

Practically all sounds used in music can be interpreted as simple sums of sine waves, an assumption used throughout this chapter. Therefore, Fourier analysis, probably the most commonly used signal analysis technique, is our basic tool for finding the spectrum of a signal. Simple steady waveforms such as square and sawtooth waves can be easily analyzed using the Fourier series method. Moreover, we will find that we can extend this method to analyze the spectra of a much wider variety of sound signals, including complex sounds produced by acoustical musical instruments and voice. However, we will defer a discussion of this generalized method, called "time-variant spectrum analysis", until Chapter 4.

Analysis is very useful, though not necessary for sound synthesis. Generally, we wish to find synthesis techniques which make it easy to produce a wide variety of interesting sounds, and such techniques are at the heart of any synthesis system. From the analysis of a musical instrument we can discover features of its sound-producing mechanism and output signal that are worthy of emulation. However, it is not necessary that we emulate its acoustics exactly. Additive synthesis, subtractive synthesis, and frequency modulation are examples of techniques which have been successfully used in ways having no counterpart in acoustical instruments. Even so, intelligent use of these techniques is enhanced by a knowledge of instrument acoustics and signal analysis. This is even more true when speaking of the design of completely new synthesis methods.

This chapter is devoted to discussing methods of constant waveform synthesis for both analog and digital systems. We should keep in mind, however, that while these methods are efficient, they are over-simplistic for most synthesis applications. Many of the attributes of timbre we are accustomed to hearing are missing in sounds produced by these static synthesis methods. For example, it is perceptually important for the spectra of acoustic instruments to be constantly changing with time; i. e., the amplitudes of the frequency components should not follow the same pattern. Another attribute results from inharmonicity of partials: While most wind instruments, voice, and bowed string tones are harmonic (periodic) to a considerable degree, the sounds of plucked and struck strings and percussion instruments are *inharmonic* (i.e., their partial frequencies do not relate to their fundamental frequencies by integral ratios) in varying degrees.

We shall construct a family of simple waveforms for analog synthesis and derive the spectra for these waveforms. However, these waveforms are not generally adequate for digital synthesis, because of the time quantization problem, which can introduce "extra frequencies" into the intended signal. Signals which are appropriate for digital synthesis are *bandlimited*, and we will discuss using sine wave summation and window pulse functions to create such signals. Further, we will delve into additive synthesis, subtractive synthesis, and envelope generation.

3.1 Introduction to Fixed Waveform Synthesis

In **fixed-waveform synthesis** we assume that the waveform is fixed for the duration of a note. However, the waveform's amplitude and frequency may vary over its duration, and the sound quality (waveform or

spectrum) may vary from one note to the next. We also assume that the signal consists solely of components whose frequencies are harmonically related to the fundamental.

A classical fixed-waveform method used with analog synthesizers is **subtractive synthesis**, where a simple waveform rich in harmonics is applied to a fixed filter. An amplitude envelope can be applied to the signal, and the frequency can be varied to produce vibrato, portamento, or glide effects. The waveform will vary automatically as a function of pitch due to the effect of the filter. Subtractive synthesis can also be accomplished in the digital domain using bandlimited waveforms and digital filters.

Digital wave generators can also be designed to load and access a large number of different waveforms of arbitrary shape, thus eliminating the necessity for a digital filter to vary the sound quality. The tradeoff between subtractive synthesis and massive waveform storage is simply the common one of digital computation speed vs. memory.

A block diagram for a generalized fixed-waveform synthesizer is shown in Figure 3-1. We will first discuss methods for producing waveforms. We will then discuss the properties of filters (both digital and analog), and typical envelope generator control functions.

Much of the following discussion in this chapter assumes that the reader is familiar with the theory of linear systems. For a more complete treatment, the reader should refer to a linear systems text such as those by Cooper & McGillem [1967] or Gabel & Roberts [1973]. This chapter is based on a review of certain well-known results of linear system theory.

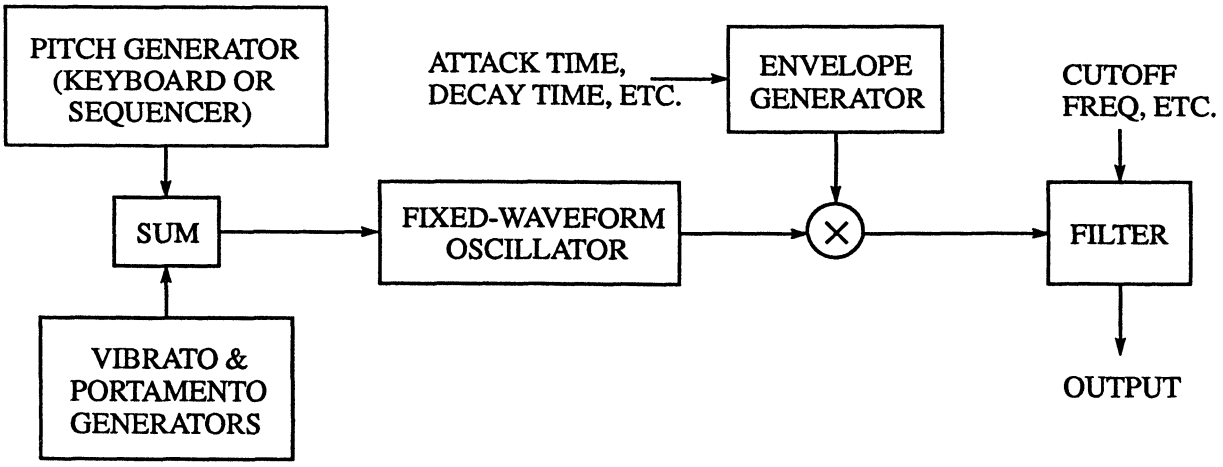


Figure 3-1. Fixed waveform synthesizer with variable pitch (fundamental frequency), envelope, and filter characteristic.

3.2 Analysis of Periodic Signals

A fixed-waveform signal is a periodic signal with constant frequency and amplitude. A good place to start in understanding periodic signals is through classic Fourier series analysis.

A signal is *periodic* if for some time interval T (called the period) the signal function s(t) satisfies

$$s(t) = s(t + T). \qquad [3.1]$$

In this case, a set of values $\{c_k\}$ (the harmonic amplitudes) and $\{\theta_k\}$ (the harmonic phases) can be found which completely characterize the signal. The signal can be completely recreated from these values using the formula

$$\hat{s}(t) = \sum_{k=0}^{\infty} c_k \cos(2\pi kt/T + \theta_k) \qquad [3.2a]$$

where $f_k = k f_1 = k/T$ is the kth harmonic frequency and c_k is the kth harmonic amplitude.
For each k, we first compute

$$a_k = \frac{2}{T} \int_{-T/2}^{T/2} s(x) \cos(2\pi kx/T)dx \qquad \text{and}$$

$$b_k = \frac{2}{T} \int_{-T/2}^{T/2} s(x) \sin(2\pi kx/T)dx \qquad [3.2b]$$

and then $$c_k = \mathbf{sqrt}(a_k^2 + b_k^2) \quad \text{and } \theta_k = \mathbf{-atan}(b_k/a_k) \qquad [3.2c]$$

The more compact *complex* definition of the Fourier series using the complex amplitude $\tilde{c}_k$ is often easer to work with in derivations:

$$\tilde{c}_k = \frac{1}{T} \int_{-T/2}^{T/2} s(x)\, e^{-j2\pi kx}\, dx, \qquad [3.3a]$$

and it is used to represent the signal according to the formula

$$\hat{s}(t) = \sum_{k=-\infty}^{\infty} \tilde{c}_k e^{j2\pi kt/T} \qquad [3.3b]$$

The real and the complex forms are related as follows:

$$2\,\tilde{c}_k = c_k e^{j\theta_k} = a_k - jb_k \qquad [3.3c]$$

$$c_k = 2|\,\tilde{c}_k\,|, \quad \theta_k = \arg(\tilde{c}_k) \qquad [3.3d]$$

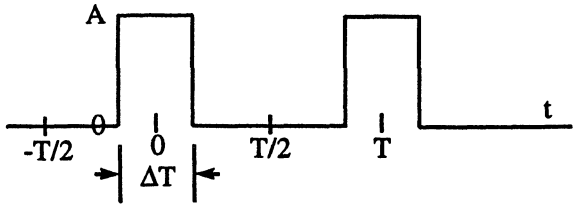
3.3 Simple Waveforms used as Sound Sources for Analog Synthesis

A number of simple waveforms are used in electronic music, in addition to the sine wave. While they are very easy to generate and work well in analog synthesis applications, their use in digital (or computer) synthesis systems is limited, due to their non-bandlimited nature. The waveforms we will treat are

1) the rectangular pulse wave
2) the impulse wave
3) the square wave,
4) the sawtooth wave
5) the asymmetrical triangle wave
6) the (symmetrical) triangle wave

The rectangular pulse is most fundamental, and spectra of the other waveforms can be derived from it.

The rectangular pulse. This wave has a flat top and bottom, a zero-to-peak amplitude A, and a pulse width of ΔT. The *duty factor* of the pulse wave is defined as the fraction of time spent on its positive cycle and is given by $\alpha = \Delta T/T$.



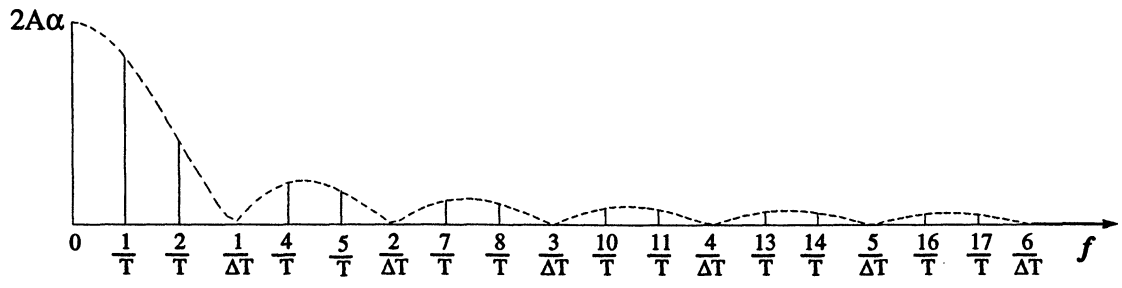
The analysis is simplified if we choose the time origin to bisect the pulse. Letting

$$\text{rect}(t,\Delta T,T) = \begin{cases} A, & -\Delta T/2 < t < \Delta T/2 \\ 0, & \Delta T/2 < |t| < T \\ \text{rect}(t+T,\Delta T,T), & \text{elsewhere} \end{cases} \qquad [3.4a]$$

we obtain the Fourier coefficient

$$a_k = \frac{2}{T}\int_{-\Delta T/2}^{\Delta T/2} A\cos(2\pi kx)dx = 2A\alpha\,\sin(\pi k\alpha)/(\pi k\alpha) = 2A\alpha\,\text{sinc}(\pi k\alpha) \qquad [3.4b]$$

Note that because of the waveform's symmetry, $b_k = 0$, and $c_k = |a_k|$. Replacing harmonic number k by frequency f_k over period (f_k/T), we can plot c_k (for $\alpha = 1/3$) as a function of frequency:



The form of this function, sin(x)/x = sinc(x), is very well known and occurs frequently in

analysis/synthesis work. A plot of this function for $\alpha << 1$ reveals that the spectrum of a rectangular pulse of short duty cycle is relatively broad band and has the following properties:

a) In gross outline the spectrum attenuates as harmonic number (or frequency) increases according to the relation 1/k or 1/f.

b) The spectrum is characterized by "zeros" which are evenly spaced and occur at harmonics $1/\alpha$, $2/\alpha$, $3/\alpha$, ... or frequencies $1/\Delta T$, $2/\Delta T$, $3/\Delta T$, ... Note that an actual zero may not occur in the harmonic spectrum if α does not divide the corresponding integer evenly.

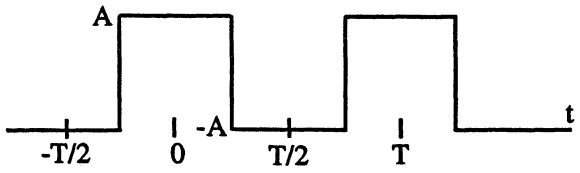
c) Beyond the lowest band of the spectrum are an infinite number of "lobes" of ever-decreasing amplitude. A rectangular pulse's spectrum does not roll off smoothly.

d) The effective bandwidth of the spectrum is given by the frequency interval up to the first zero. Thus, this bandwidth equals the inverse pulse width ($1/\Delta T$). To increase the bandwidth one simply needs to decrease the pulsewidth. However, note that the duty factor is also a multiplier on the amplitude, so that the general amplitude height of the spectrum decreases as the bandwidth is increased. It is interesting to note that although the rms amplitude decreases as the pulse width decreases, loudness does not decrease appreciably until a certain pulse width is reached. For the reason why this is true, recall our theory of loudness summation in Chapter 2.

e) If we increase the repetition period to infinity, a single pulse results (time domain), and the frequency components become infinitesimally close to form a continuous spectrum (Fourier transform). However, this does not change the position of the zeros or the general shape of the spectrum. **The envelope of the spectrum for a single finite-width pulse has the same shape as that of the periodic pulse train.**

The Impulse Wave. This theoretical waveform is approached as the pulse width is reduced to zero. The spectrum of the impulse consists of a series of equal-amplitude components at frequencies f_1, $2f_1$, $3f_1$, ... (where $f_1 = 1/T$). It is a very convenient waveform for analysis of filters and reverberators, but it is impossibile in the real analog world. However, in the digital world an impulse is simply a single, isolated "on" sample surrounded by "off" samples and thus is a very practical as well as theoretical tool.

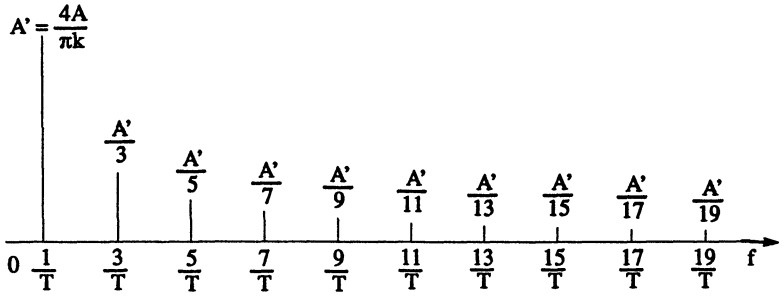
The Square Wave. This waveform has been very often used in electronic music. Its peak-to-rms amplitude ratio is 1:1, it has perfect odd symmetry, and its spectrum contains only odd harmonics (i. e., f_1, $3f_1$, $5f_1$, ...) rendering its sound "hollow" or "clarinet-like".



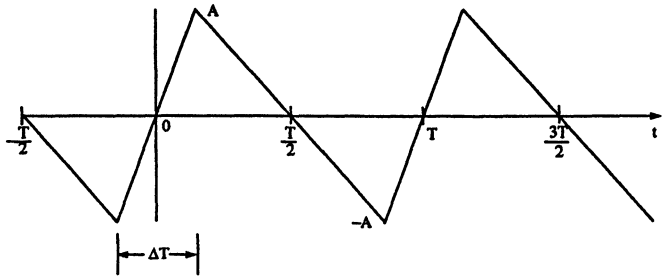
Also, its harmonic amplitudes are inversely proportional to harmonic number or frequency. Letting the peak-to-peak amplitude of the square wave be 2A, so that it alternates between -A and A, we can derive the Fourier coefficients from Equation 3.4a by replacing A by 2A and the duty factor α by 1/2. This results in:

$$a_k = \begin{cases} \frac{4A}{\pi k}(-1)^{(k-1)/2}, & k \text{ odd} \\ 0, & k \text{ even} \end{cases} \qquad [3.5]$$

The sign alternates on the odd harmonics, and the even harmonics are zero because $\sin(\pi k/2) = 1$ for k = 1, 0 for k = 2, -1 for k = 3, 0 for k = 4, etc. After taking the absolute value, the spectrum looks like



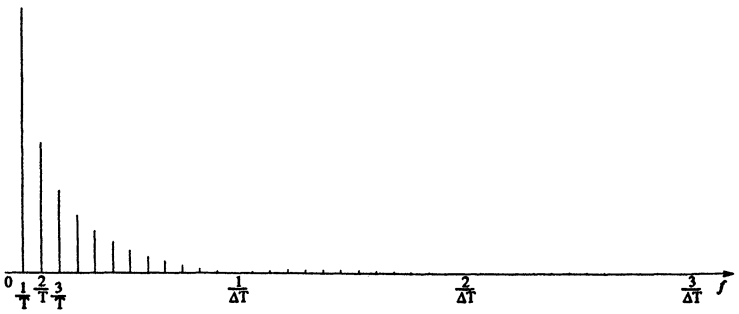
The Asymmetrical Triangle Wave. We could work this out from scratch by setting the wave up as an odd function and using Equation 3.2b.



Centering the origin of analysis in the center of the upward slope of the triangle wave, we would find that $a_k = 0$. As an alternative, we can derive the Fourier series by thinking of the triangle wave as the integral (or "anti-derivative") of the rectangular pulse wave. (It is the wave whose derivative yields the rectangular pulse.) Either way, we will arrive at

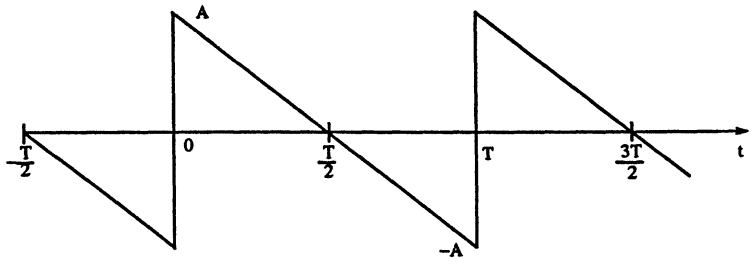
$$b_k = \frac{2A}{\alpha(1-\alpha)} \frac{\sin(\pi k \alpha)}{\pi^2 k^2} \qquad [3.6]$$

Again, after taking absolute valuse, the spectrum (for $\alpha = 1/13$) looks like:



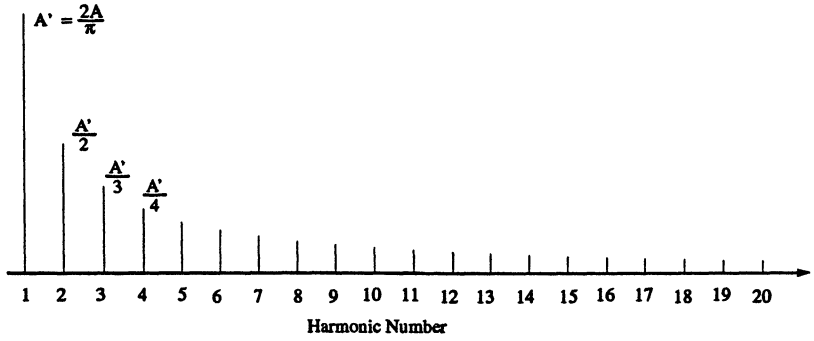
This spectrum has similar properties to the rectangular pulse but rolls off much faster. The overall rate is ~ $1/k^2$ (-12 dB/8va), as compared to ~ 1/k (-6 dB/8va) for the rectangular pulse. Otherwise, the spectral properties listed for the rectangular pulse given above apply, e.g., the zeros of the spectrum are in exactly the same places. This is what we would get if we derive the complex coefficient of the asymmetrical triangle wave from the coefficient of the pulse wave by dividing the latter by $j2\pi k$. A proof of this method will be given at the end of this section.

Sawtooth Wave.



Its spectrum may be derived from the result for the asymmetrical triangle wave by taking the limit as $\alpha \rightarrow 0$. This gives:

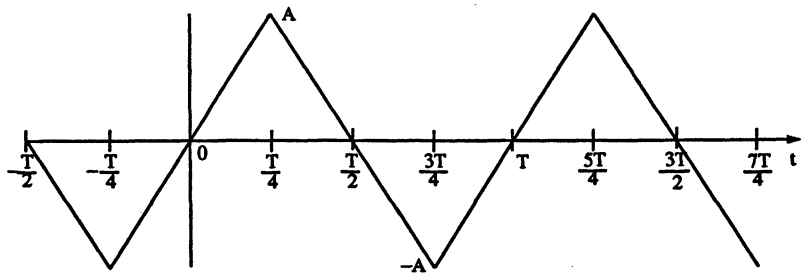
$$b_k = \frac{2A}{\pi k} \qquad [3.7]$$



The result is similar to that of the square wave. In both cases the harmonic amplitude is inversely proportional to harmonic number. However, whereas the even partials are missing for the square wave, the sawtooth has all partials present. This waveform is very appropriate as a source waveform for subtractive synthesis in analog systems because it is typical of acoustic excitation waveforms -- e. g., it resembles the force waveform at the bridge of a bowed string instrument and the glottis vibration waveform of the human vocal chords. Also, it contains appreciable spectral energy in successive octave bands (the rms amplitude of each successive band decreases only as $(1/f)^{1/2}$), and it has a low peak-to-rms ratio (therefore, not playing havoc with system dynamic range).

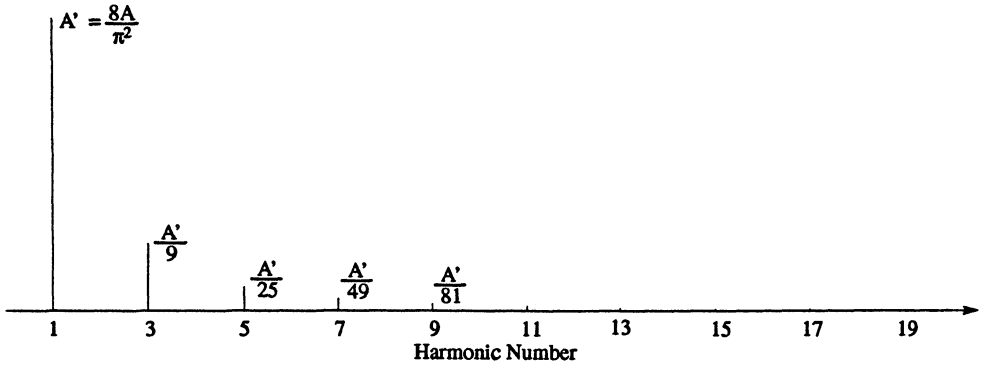
Note that we can also derive Equation 3.7 by considering the sawtooth as the integral of the impulse wave and by using the method given at the end of this section.

The (Symmetrical) Triangle Wave.



The spectrum of this waveform can be derived either as the integral of a square wave (see Section 3.4) or by taking $\alpha \rightarrow 0.5$ for the assymetrical triangle wave result of Equation 3.6. Either way we arrive at

$$b_k = \begin{cases} \frac{8A}{\pi^2 k^2}(-1)^{(k-1)/2}, & k \text{ odd} \\ 0, & k \text{ even} \end{cases} \qquad [3.8]$$



Like the square wave, the triangle wave consists only of odd partials with alternating sign (with respect to all sine waves). However, the spectrum rolls off as $1/k^2$ (-12 dB/8va) instead of $1/k$ (-6 dB/8va). The triangle wave is a rather dull sounding waveform and it is not as useful as a source waveform for subtractive synthesis as are the sawtooth, pulse, and square waves. It is used quite often as a waveform to be shaped into a sine wave by nonlinear distortion (otherwise known as waveshaping or function generation). It also makes, as it is, a rather crude "poor man's sine wave".

3.4 Useful Relationships Between the Fourier Series and the Fourier Transform

Fourier transforms are often used for non-periodic signals, i.e., signals which are either of short duration (*time-limited*) or are changing in a statistical manner. For the special case of a time-limited signal, the Fourier series can be thought of as the Fourier transform sampled in the frequency domain, and the Fourier transform can be considered to be the interpolated *envelope* of the Fourier series. Also, if we make the fundamental frequency of a Fourier series approach zero and view the spectrum as a function of frequency, the Fourier series will approach the transform in the limit. If we let s(t) be a signal which is non-zero only on the interval $|t| < T/2$, the Fourier transform is

$$S(j\omega) = \int_{-\infty}^{\infty} s(t)\, e^{-j\omega t}\, dt = \int_{-T/2}^{T/2} s(t)\, e^{-j\omega t}\, dt \qquad [3.9a]$$

Because of the resemblance of this result to the definition of the complex Fourier series coefficient (see Equation 3.3a), we can state

$$\tilde{c}_k = \frac{1}{T} S(j2\pi k/T) \qquad [3.9b]$$

In other words, the Fourier series coefficients are equal to values of the Fourier transform sampled at **harmonics** of the fundamental $2\pi/T = 2\pi f_1$ and weighted by 1/T. Conversely, passing a smooth curve through the discrete $|\tilde{c}_k|$ values will approximate the Fourier transform's magnitude. We call this smooth curve the *spectral envelope* of the discrete spectrum.

Moreover, here is another very useful result from Fourier theory. If g(t) and f(t) are related by

$$g(t) = \int f(t)dt, \qquad [3.10a]$$

then their Fourier transforms are related by

$$G(j\omega) = F(j\omega)/j\omega\,. \qquad [3.10b]$$

In the case where Fourier series are applicable this translates into

$$G_k = T F_k/(j2\pi k), \qquad [3.10c]$$

where G_k and F_k are the complex Fourier series coefficients of g(t) and f(t), respectively.

Because of the **j** operator in the denominator, this operation translates an a_k (cosine-oriented) real coefficient into a b_k (sine-oriented) one.

From the integral relationship given by 3.9a,c, there follows a general rule about the spectral rolloff of waveforms: *If n derivatives are required to produce an impulse, the waveform's spectrum will ultimately roll off as* $\sim\kappa/k^n$, for some constant κ.

3.5 **Bandlimited Waveforms for Digital Synthesis**

In digital systems continuous signals are approximated as a series of quantized samples which are spaced apart in time at regular intervals. The samples occur at a constant rate, called the *sample rate* or *sampling frequency*. The sampling theorem states that a signal cannot be properly represented in terms of its samples -- i.e., the original or intended signal cannot be reconstructed, even theoretically -- unless the following condition is satisfied:

$$\text{sample frequency} > 2\times(\text{highest frequency of the signal}) \qquad [3.11a]$$

According to this theorem a signal can be perfectly reconstructed if its highest frequency component is less than half of the sample frequency. For a fundamental frequency f_1, highest harmonic K, and a sample frequency = f_s , this condition is

$$K f_1 < f_s/2 \qquad [3.11b]$$

where $f_s/2$ is known as the *Nyquist frequency*.

The sampling theorem can be illustrated in terms of how an audio spectrum is transformed by the sampling process. Figure 3-2 compares an audio spectrum with the same spectrum after it has been sampled. Note the sideband below the sampling frequency, which is a mirror image of the original audio spectrum. When the tail end of this lower sideband extends to the left of the half-sample frequency (as shown in Figure 3-2c), it mixes with the original spectrum components. This effect is sometimes referred to as *aliasing* or *foldover*.

Violation of Equation 3.11b can result in distorted, sometimes raucous sounds. In terms of spectra, this can be explained as a kind of *heterodyning* of the partials with the sample frequency. I.e., corresponding to the kth harmonic, the frequency $f_S - kf_1$ appears in the output signal. As long as this frequency exceeds kf_1, the difference component can be easily removed by filtering it with a sharp-cutoff low pass filter. Otherwise, unless there is *a priori* knowledge of k, f_1, and f_s, extracting the difference component from the principal component is impossible. If the foldover frequency corresponds to one of the original frequencies, its effect may not be noticed. This happens when f_1 is an integral divisor of f_s. But, in all probability, the frequencies of difference components will be different from the frequencies of any of the

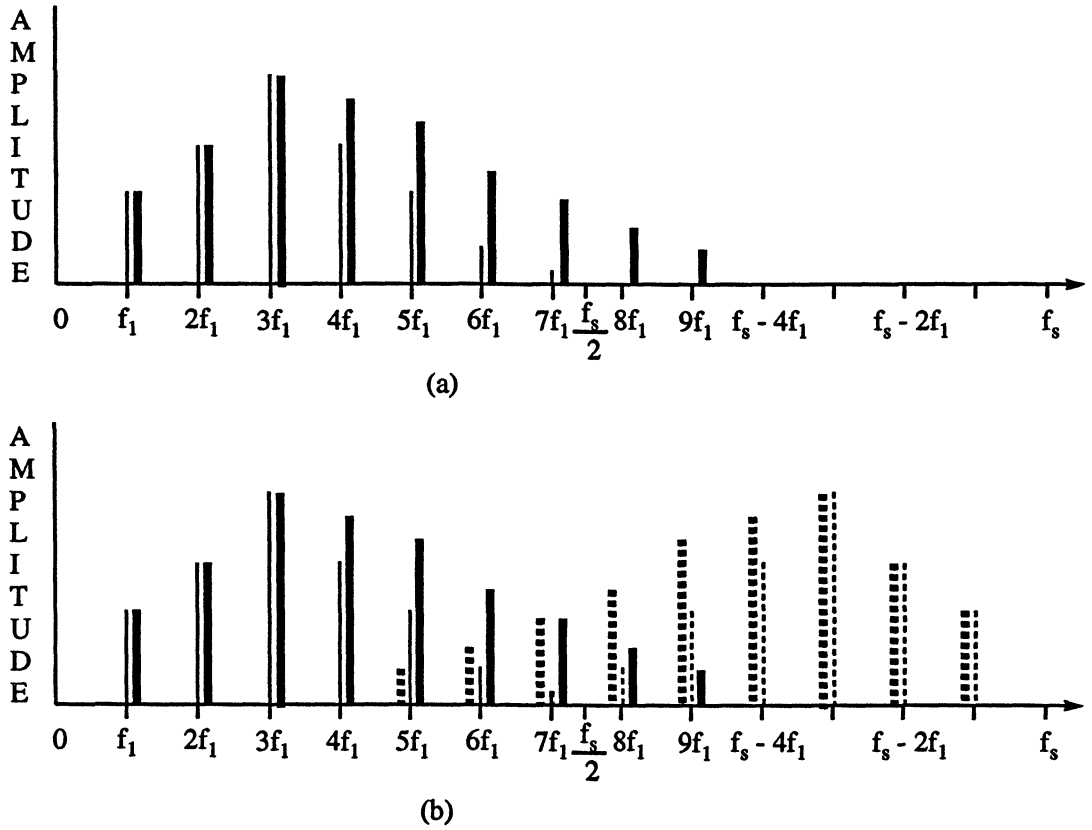


Figure 3-2. (a) Original unsampled audio spectrum: Thin vertical lines represent a spectrum bandlimited to $f_s/2$; fat lines represent a modification to this spectrum which violates the Nyquist limint. (b) Signal's spectrum after sampling has taken place: Dotted thin and fat vertical lines represent the aliasing components corresponding to the respective original audio components. Note that the aliasing components intertwine with the components of the original signal when $f < f_s/2$.

principal, desired components, and thus the difference components will usually be painfully audible. The only way to avoid this problem -- the problem of "folded over components" -- is to generate waveforms which have essentially zero amplitude for frequencies at and above the half-sample frequency. Signals which have zero spectral amplitude above a certain frequency are called **bandlimited waveforms.**

Three ways to produce bandlimited (or nearly bandlimited) waveforms are

a) Scanning a table prestored with a sum of harmonic sine waves.

b) Computation with a summation formula, a simplified formula for computing a finite sum of sine or cosine waves.

c) Scanning a table prestored with a special "window function". The scanning operation alternates with generation of "dead space", an interval of zero value. The result of this process is a nearly bandlimited periodic signal.

3.5.1 Fixed Waveform Additive Synthesis

There are two types of additive synthesis used in electronic/computer music. One form assumes that the amplitudes and frequencies (and perhaps the phases) of all components are time-varying and are controlled independently. We will be covering this elaborate case in Chapter 4. For the other type, a fixed waveform is created by adding harmonics with arbitrary amplitudes and phases for a single period which is repeated indefinitely. This is our subject here. The general formula for fixed-waveform additive synthesis is

$$\hat{s}(t) = \sum_{k=1}^{K} c_k \cos(2\pi k f_1 t + \theta_k) . \qquad [3.12]$$

In a computer one can add up the individual sinusoids to form a table using an algorithm (written in C) such as

```
pi2 = 8.*atan(1.);              /* twice the number PI */
for (i = 0;i<I;i++){
   sum = 0;
   fac = (pi2/I)*i;
   for (k=1;k<=K;k++){
      sum = sum + c[k]*cos(k*fac + theta[k]);        /* adding up the Fourier series */
   }
   s[i] = sum;          /* the table entry */
}                                                                              [3.13]
```

where I is the number of table entries or samples in a period and K is the number of harmonics in the waveform.

For a sample frequency f_s and fundamental frequency f_1 the largest value of K (giving the maximum number of harmonics) is $K = .5f_s/f_1$. This means that wavetables for high-pitched fundamentals may contain only a few harmonics, whereas a larger number of harmonics are possible for low fundamentals.

For example, if $f_s = 20000$ and $f_1 = 4000$ Hz, only 2 harmonics are possible, but for $f_1 = 100$ Hz we can use 99 harmonics. To get around this problem, we can adopt a strategy to recompute the wave table each time a new pitch is to be generated, or store tables for all possible values of K, or use some compromise between the two possibilities.

3.5.1.1 Equal-Amplitude Sums of Sines: Effect of Harmonic Phases

A special version of Equation 3.12 results from setting $c_k \equiv 1$:

$$s(t) = \sum_{k=1}^{K} \cos(2\pi k f_1 t + \theta_k), \qquad [3.14]$$

where the $\{\theta_k\}$ is a set of arbitrary phase values. The rms amplitude is $\sqrt{K/2}$, regardless of the phases.

An interesting problem is to determine a set of phases which minimizes the waveform's peak value, which is defined as

$$P = \max(\text{abs}(s(t))) \text{ for } 0 \le t \le T \qquad [3.15]$$

The virtue of minimizing P is that for a limited amplitude range the signal of the greatest power (and loudness) can be generated. Recall that the ear is insensitive to phase (unless the fundamental frequency is very low). So we might as well produce maximum power to the output, in order to minimize signal-to-noise ratios, head-room problems, etc. Choosing all phases to be zero gives the worst case, i.e., the greatest peak value. In this case, all of the peaks of the individual cosines line up, and so for n harmonics the peak value (max abs value) is n. For a given number of harmonics there is no simple way to find the optimum phases to minimize P, but there are a couple of good starting points. One is to use random phase values. This results in a waveform which is random-looking but which repeats with frequency f_1. The other is use "Schroeder phases" [Schroeder, 1970], which are given by

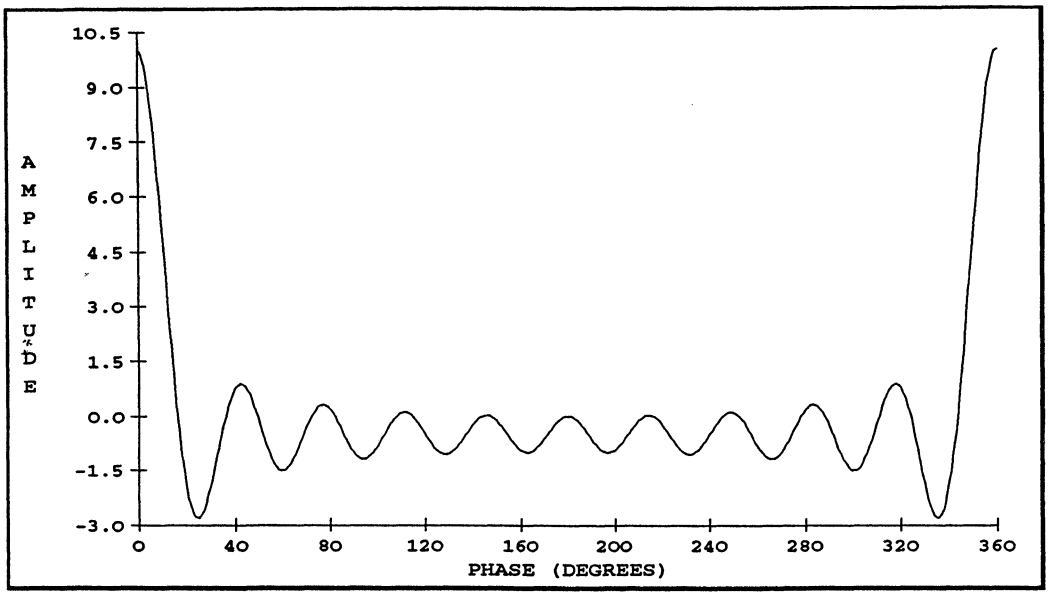
$$\theta_k = \frac{\pi}{K} k^2 + \theta_0, \qquad [3.16]$$

where θ_0 is arbitrary. It turns out that using phases according to Equation 3.16 yields a waveform that resembles a sine wave frequency-modulated by a ramp function.

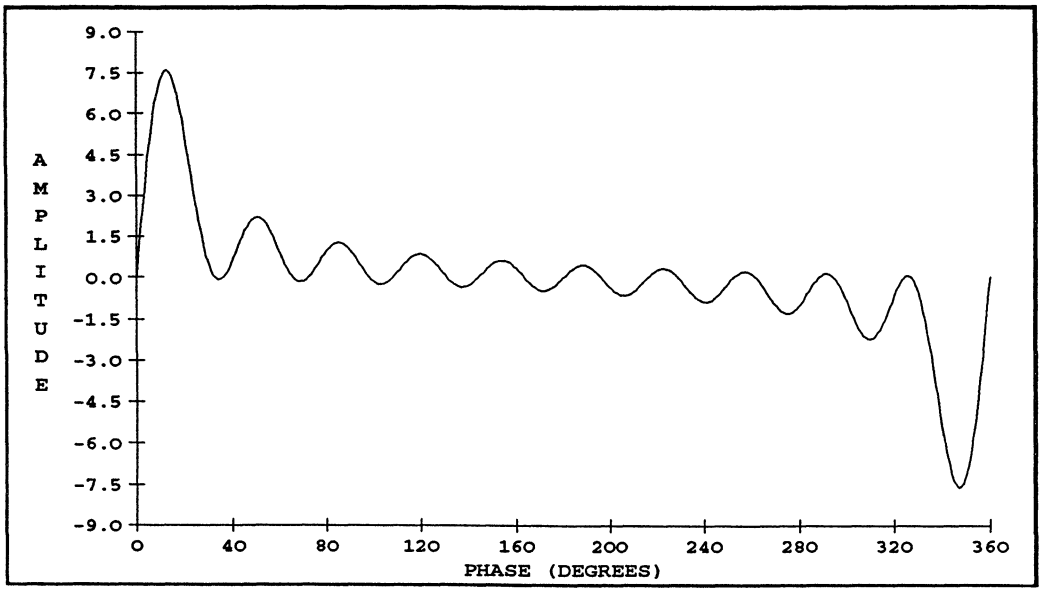
Taking an example of 10 harmonics Figure 3.3a shows the sum waveform for all phases set to zero (all cosines), and Figure 3.3b shows the waveform for all sines (all phases set to $-\pi/2$). Random phases were used to generate the waveform of Figure 3.4a, and Schroeder phases for Figure 3.4b. The peak values for the four cases are:

All cosines: 10. All sines: 7.59 Random phases: 4.48 Schroeder phases: 3.93

These waveforms will all sound the same if the fundamental frequency is above 100 Hz or so. However, below that frequency they will actually begin to sound different. When the repetition rate is suitably low, the ear becomes sensitive to the harmonic phases and the detailed time behavior of the sine wave sums.

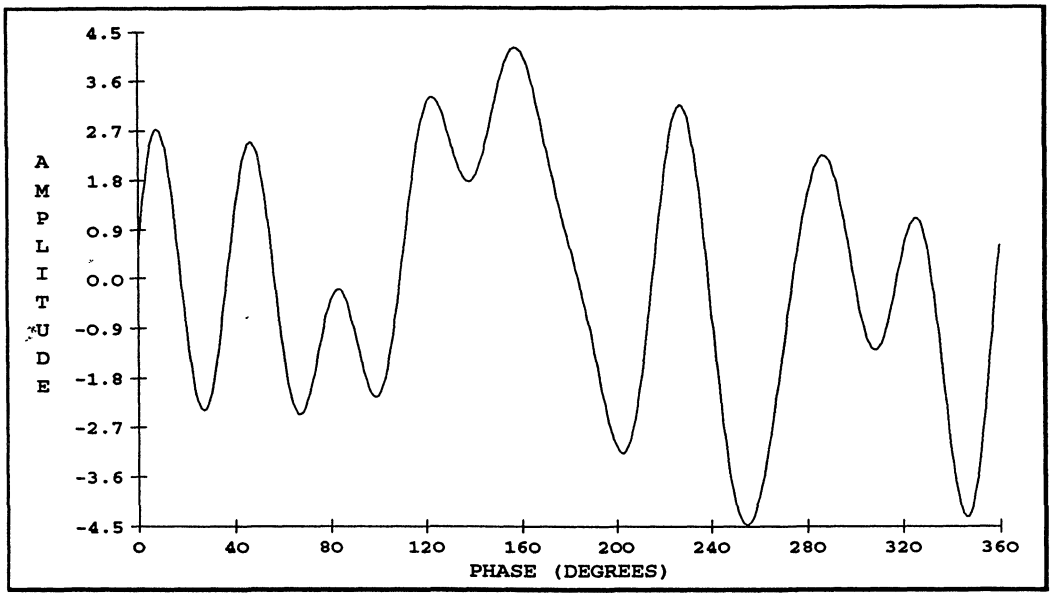


(a)

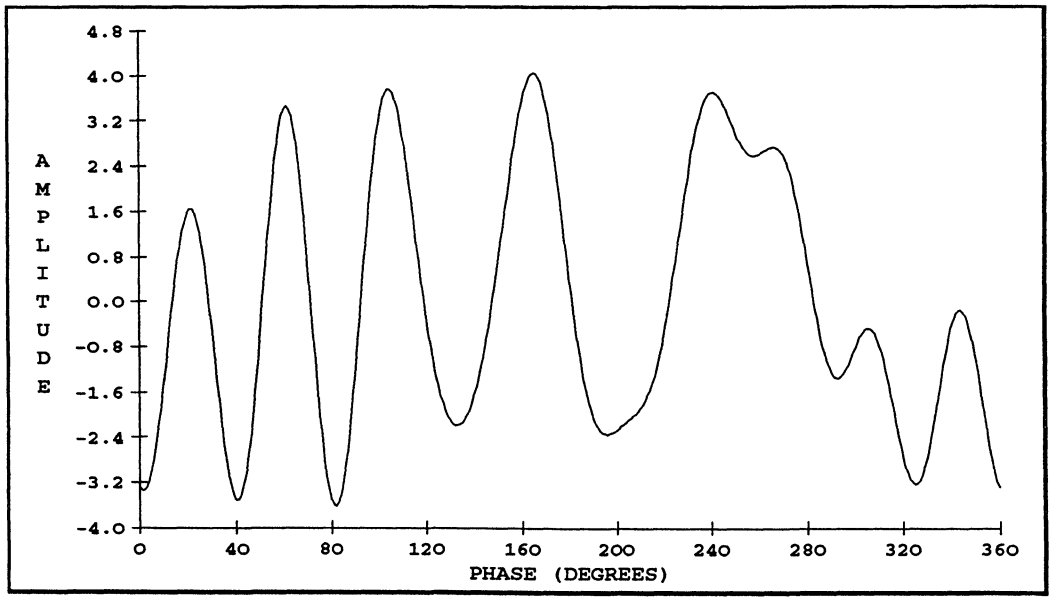


(b)

Figure 3-3. Bandlimited pulse waves: a) 10 equal-amplitude harmonics using cosine components. b) 10 equal-amplitude harmonics using sine components.



(a)



(b)

Figure 3-4. Bandlimited waveforms consisting of 10 equal-amplitude harmonics using
a) random phases (degrees): 220, 63, 262, 193, 312, 67, 234, 317, 324, 271;
b) Schroeder phases: 148, 202, 292, 58, 220, 58, 292, 202, 148, 130.

Over a range of 2 <= K <= 40, the peak amplitude using Schroeder phases hovers between 1.2 $K^{-.5}$ and 1.53 $K^{-.445}$. Since the rms amplitude of the sum of n harmonics, is .707 $K^{-.5}$ regardless of the phases used, the peak-to-rms ratio for the Schroeder phase case is approximately 1.75 to 2.0. By using "genetic algorithm-based optimization programs, it is possible to improve on this significantly -- ratios as low as 1.38 have been achieved (Horner and Beauchamp, 1996). Note that the ratio for a sine wave is 1.4142. No one knows what the ultimate minimum ratio is.

3.5.2 **Summation Formulas**

It is possible to express some sinusoid sums in more compact, easier-to-compute forms. Starting with the geometric series

$$\sum_{k=1}^{K} r^k = r(1 - r^K)/(1 - r) \qquad [3.17a]$$

and the identity

$$\sum_{k=1}^{K} \cos(k\varphi) = \mathbf{Re}\{\sum_{k=1}^{K} e^{jk\varphi}\}, \qquad [3.17b]$$

we can derive

$$\sum_{k=1}^{K} \cos(k\varphi) = .5\{ \sin[(K+.5)\varphi])/\sin(.5\varphi) - 1\} \qquad [3.17c]$$

Similarly, using **Im** instead of **Re** in the above equation, we arrive at

$$\sum_{k=1}^{K} \sin(k\varphi) = .5\ \{\cos[.5\varphi] - \cos[(K+.5)\varphi]\}/\sin(.5\varphi) \qquad [3.17d]$$

Equations 3.17c, 3.17d are summation formulas for computing bandlimited waveforms for a particular number of harmonics K. They cut down considerably on the amount of computation required to generate a sum of sine waves if K is large. Graphs of these "all-cosine" and "all-sine" pulses for the case K=10 were shown in Figures 3.3 a & 3.3b. Note that they have ripples which extend over the entire periods of the waveforms -- the number of ripples is equal to the number of harmonics in the sum.

Integration of equation 3.17c with respect to φ leads to a formula for a "bandlimited sawtooth". We will leave it to the reader to see how this works out.

The resulting waveforms make excellent excitation signals for subtractive synthesis (i.e., spectral modification with linear filters) in a digital system, since they are inherently bandlimited. Equation 3.17c, for example, can be implemented as two sine table lookups, a divide, and a subtract operation. This type of pulse synthesis was a regular feature of the Systems Concepts Digital Synthesizer [Sampson, 1980] and the Portable Digital Sound Synthesis System [Alles, 1977]. For further discussion see Dodge and Jerse [pp. 149-153, 1985].

3.5.3 Nearly-Bandlimited Window Pulse Functions

Window functions are special functions which, like the rectangular pulse, are "time-limited, but are smoother than the rectangular pulse and thus are approximately bandlimited. A repeating window function pulse can be used as a harmonic-rich source waveform if it is generated once every period while the remainder of each period is filled with "dead space" or value zero. For an *effective pulse width* of ΔT, the bandwidth will remain approximately $1/\Delta T$ regardless of the period $T = 1/f_1$. When the pulse function is sampled at rate f_s , it is only necessary to keep $1/\Delta T$ sufficiently below $1/2f_s$, and both f_s and ΔT can be kept fixed over a wide range of performance frequencies f_1. Note, however, as in the case of the spectrum of the rectangular pulse (given by Equation 3.4b), the amplitude (or "strength") of the spectrum of a finite pulse wave is inversely proportional to its period, and thus must be compensated for as its frequency changes. (The idea of using window pulses for subtractive synthesis was introduced by Bass and Goeddel [1981].) Figure 3-5 shows a typical window pulse waveform.

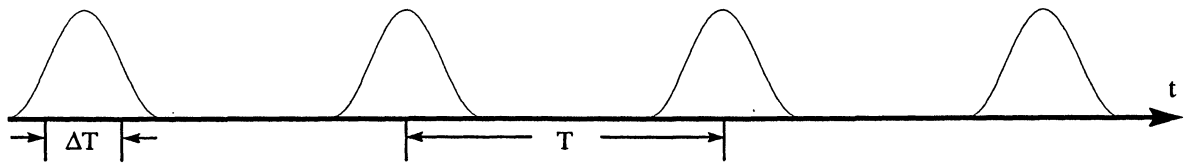


Figure 3-5. A window pulse waveform. The effective pulse width is ΔT, whereas the physical pulse width is larger. The distance between pulses is the period T, which is the inverse of the fundamental frequency f_1. The waveform is zero in the "dead space" between the pulses.

To achieve the bandlimited behavior we want, we will consider window pulse functions of the form

$$w(t) = \begin{cases} \frac{1}{m+1} + \sum_{p=1}^{m} \alpha_p \cos(2\pi pt/(m+1)\Delta T), & |t| < (m+1)\Delta T/2 \\ 0, & |t| > (m+1)\Delta T/2 \end{cases} \qquad [3.18]$$

where m is the "order" of the window function, $(m+1)\Delta T$ is the physical width of the window, and ΔT is the *effective* window width. Note that m=0 corresponds to the rectangular pulse.

Probably the easiest window function to remember is the **hanning window** (m=1) which is given by

$$\text{hann}(t;\Delta T) = \begin{cases} \cos^2(\pi t/2\Delta T) = .5 + .5\cos(\pi t/\Delta T), & |t| < \Delta T \\ 0, & |t| > \Delta T \end{cases} \qquad [3.19a]$$

Like the rectangular pulse, the first zero of the Fourier transform of hann(t;ΔT) occurs at $f = 1/\Delta T$; however, compared to the rectangular pulse the magnitude of the hanning's transform is much reduced beyond that point. It is reduced even more if the second coefficient is changed from .5 to .426. In this case the function becomes the **Hamming window** (m=1):

$$\text{hamm}(t;\Delta T) = \begin{cases} .5 + .426\cos(\pi t/\Delta T), & |t| < \Delta T \\ 0, & |t| > \Delta T \end{cases} \qquad [3.19b]$$

The physical widths of the hanning and Hamming windows are $2\ \Delta T$.

A further refinement is the **Blackman-Harris window** [Harris, 1978] (m=3), which is given by

$$\text{blah}(t;\Delta T) = \begin{cases} .25+.3403\cos(\pi t/2\Delta T)+.0985\cos(2\pi t/2\Delta T)+.0081\cos(3\pi t/2\Delta T), & |t| < 2\Delta T \\ 0, & |t| > 2\Delta T \end{cases} \quad [3.19c]$$

The physical width of the Blackman-Harris window is 4 ΔT.

The Fourier series for the pulse train which results from using any one of these window pulse functions can be obtained by first evaluating the Fourier transform of the corresponding time-limited isolated, solitary pulse and then using the result of Equation 3.9b:

$$\tilde{c}_k = \frac{1}{T} W(j\omega_k), \text{ where } \omega_k = 2\pi k/T = 2\pi k f_1, \text{ and where}$$

$$W(j\omega) = \int_{-\infty}^{\infty} w(\tau)\, e^{-j\omega\tau}\, d\tau = \int_{-(m+1)\Delta T/2}^{(m+1)\Delta T/2} w(\tau)\, e^{-j\omega\tau}\, d\tau \quad [3.20a]$$

By substituting the formulation for w(t) of Equation 3.18 into Equation 3.20a, expanding cos() in terms of complex exponentials, and interchanging integration and summation, we can arrive at:

$$W(j\omega) = \int_{-(m+1)\Delta T/2}^{(m+1)\Delta T/2} \{\frac{1}{m+1} + \sum_{p=1}^{m} \alpha_p \cos(2\pi pt/(m+1)\Delta T)\} e^{-j\omega\tau}\, d\tau$$

$$= \Delta T\, \text{sinc}(.5\omega(m+1)\Delta T)$$

$$+ \sum_{p=1}^{m} .5(m+1)\Delta T\alpha_p \{\text{sinc}(.5(m+1)\Delta T - p\pi) + \text{sinc}(.5(m+1)\Delta T + p\pi)\} \quad [3.20b]$$

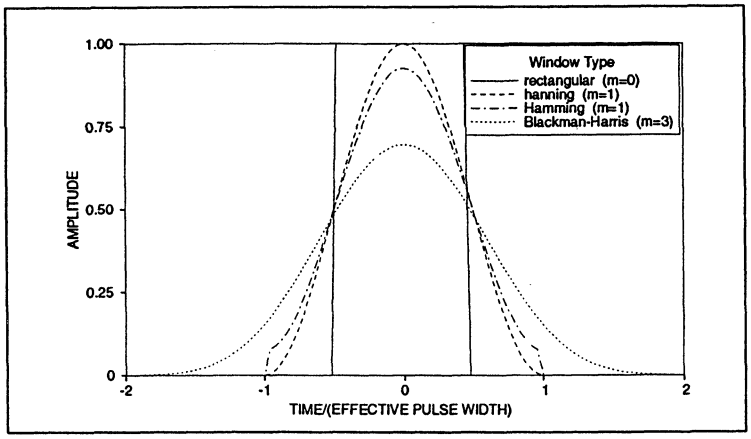
For each of the three cases discussed above, we can now write down their Fourier transforms:

$$\text{HANN}(j\omega) = \Delta T \{ \text{sinc}(\omega\Delta T) + .500 [\text{sinc}(\omega\Delta T - \pi) + \text{sinc}(\omega\Delta T + \pi)] \} \quad [3.20c]$$

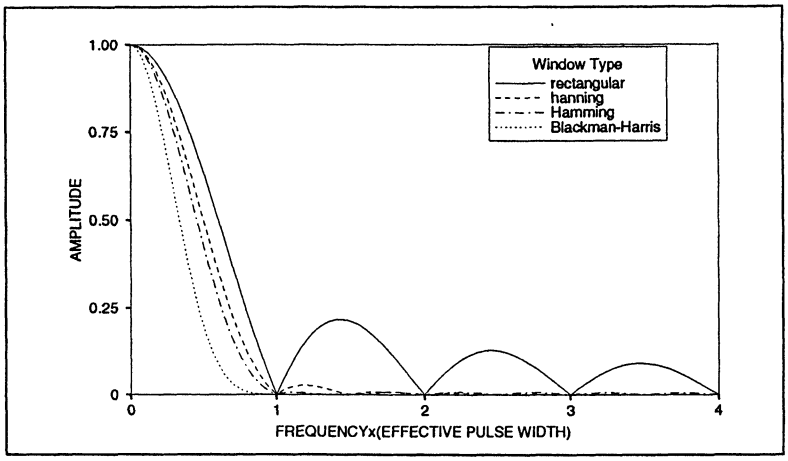
$$\text{HAMM}(j\omega) = \Delta T \{ \text{sinc}(\omega\Delta T) + .426 [\text{sinc}(\omega\Delta T - \pi) + \text{sinc}(\omega\Delta T + \pi)] \} \quad [3.20d]$$

$$\begin{aligned} \text{BLAH}(j\omega) = \Delta T \{ & \text{sinc}(2\omega\Delta T) + .6805 [\text{sinc}(2(\omega\Delta T - \pi/2) + \text{sinc}(2(\omega\Delta T + \pi/2)] \\ & + .1969 [\text{sinc}(2(\omega\Delta T - \pi) + \text{sinc}(2(\omega\Delta T + \pi)] \\ & + .0163 [\text{sinc}(2(\omega\Delta T - 3\pi/2) + \text{sinc}(2(\omega\Delta T + 3\pi/2)] \end{aligned} \quad [3.20e]$$

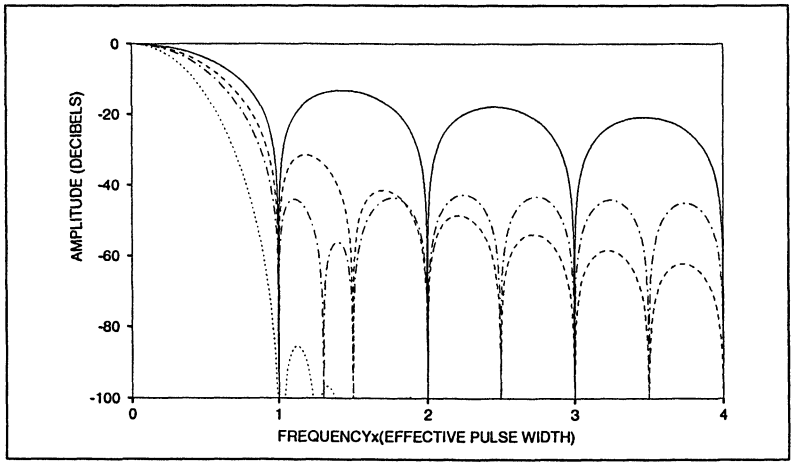
Figure 3-6 compares the window pulse shapes and Fourier transforms of the four different window functions. Compared to the rectangular window which has a maximum sidelobe of about -13 dB, the Hamming window's transform is reduced below -43 dB for frequencies above $1/\Delta T$. Also, we see that the transform of the Blackman-Harris window is better than -92 dB down for frequencies greater than this frequency. The hanning is inferior to the Hamming for $1/\Delta T < f < 2/\Delta T$ but it is superior beyond that



(a)



(b)



(c)

Figure 3-6. Comparison of window functions and their Fourier transforms: a) window pulse functions. b) magnitude of window tranforms; c) window transforms shown with decibel amplitude scale. The "effective pulse width" is the same as ΔT, as referred to in the equations in the text.

point. It also has a superior response below $1/\Delta T$. All of the transforms are zero at certain frequencies above $1/\Delta T$, namely at $f = q/((m+1)\Delta T)$ where q is an integer greater than m.

Also, $W(0) = \Delta T$ for all transforms, since all of the sinc functions except the first one are zero for $\omega = 0$. After the first zero occurs, the various sinc(x) functions cancel each other out to a considerable degree (assuming that the α_k's are adjusted properly). This gives rise to the approximately bandlimited behavior of these functions.

The spectrum of a pulse waveform with frequency f_1 is found by sampling the waveform's Fourier transform at harmonic frequencies $k f_1$, where $k = 1, 2, 3, \ldots$ To apply a reasonable frequency scale to the transform we can set the first zero frequency to the half-sample frequency or to different positions relative to that frequency depending on accuracy requirements. For example, to avoid foldover we can stipulate that all components in the spectrum of the unsampled pulse train must be below a certain amplitude when frequencies are equal to or greater than the half-sample frequency. For a given window type, this determines the pulse width of the window and consequently the spectral envelope of the pulse's spectrum. A practical limit on the fundamental frequency is the frequency at which the spectral envelope is -3 dB down from the zero frequency response. Any frequency component above this frequency will be rapidly attenuated. Another limitation is that the fundamental frequency must be less than the inverted physical pulse width. A higher frequency would demand that the period be less than the window width, an impossibility (unless the windows are overlapped). These frequency limits are tabulated below for setting window transform threshold levels of -40 dB and -60 dB at $f_s/2 = 10000$ Hz (assuming $f_s = 20000$ Hz) for each of the window types.

Window Type	Maximum Fundamental Frequency Corresponding To		
	-3 dB freq criterion	pulse width limit	1st zero frequency
For -40 dB at 10000 Hz:			
rectangular	138 Hz	316 Hz	316 Hz
hanning	2571 Hz	3571 Hz	7142 Hz
Hamming	3368 Hz	5263 Hz	10526 Hz
Blackman-Harris	2938 Hz	3125 Hz	12500 Hz
For -60 dB at 10000 Hz:			
rectangular	14 Hz	32 Hz	32 Hz
hanning	1084 Hz	1506 Hz	3012 Hz
Hamming	154 Hz	241 Hz	482 Hz
Blackman-Harris	2611 Hz	2778 Hz	11111 Hz

Table 3.1 Comparison of the maximum fundamental frequencies that can be generated with window pulse trains when the window widths are adjusted to give the same worst-case response for $f > 10000$ Hz.

We see how terribly a rectangular pulse performs under sampled signal conditions. If a -40 dB limit is specified, to avoid foldover for $f_s = 20000$, the highest fundamental frequency must be kept below 138 Hz. Note that since the first zero frequency is set much below the half-sample frequency, the pulse's transform is decidedly not flat below the Nyquist frequency. On the other hand, if the Hamming or

Blackman-Harris window is used, the practical limit on fundamental frequency is 3368 or 2938 Hz, respectively. If we impose a more strigent limit of -60 dB, the highest practical frequency for the rectangular pulse is only 14 Hz, whereas for the Blackman-Harris window this frequency is 2611 Hz. Note that the Hamming window does poorly under this requirement. To improve the frequency range by some factor we would have to increase the sample rate by the same factor. For example, assuming the -60 dB criterion, if we would like to synthesize spectral components flat within the -3 dB criterion up to 10000 Hz using the pulse window technique, we could accomplish this with the Blackman-Harris window if the sample rate were increased to 76.6 KHz. Figure 3-7 shows the spectrum of a 500 Hz tone based on a Blackman-Harris pulse with ΔT set to .0001 sec. This easily satisfies the -60 dB (actually < -80 dB) requirement for a sample rate of 20000 Hz.

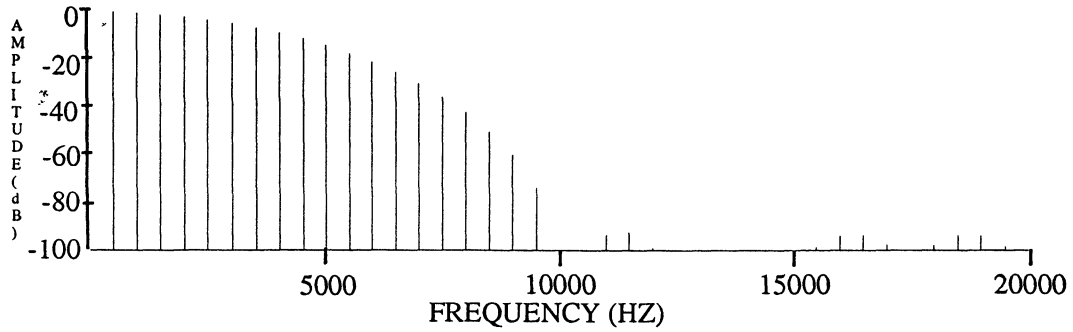


Figure 3-7. Spectrum of 500 Hz Blackman-Harris pulse wave with effective pulse width = .0001 sec and physical pulse width of .0004 sec.

Summary of Bandlimited Waveform Generation Techniques

With the fixed-waveform additive synthesis method, any arbitrary spectrum can be synthesized to a wavetable, and then the wavetable can be scanned to produce samples for an arbitrary frequency. The disadvantage of this method is that the Nyquist frequency is exceeded if the number of harmonics times the fundamental frequency is greater than the half sample frequency. The summation formula technique surmounts this problem by efficiently synthesizing only the specified number of harmonics without reloading a table. It gives good results when the number of frequencies is equal to (or a little less than) the half-sample frequency divided by the fundamental frequency. In a digital implementation the sum of cosines version requires two sine table lookups, two table index increments, a divide, and a subtract opration. Also, the increment for one of the lookups varies with the number of harmonics generated, which in turn must be calculated to keep the maximum frequency below the half samply frequency as the fundamental frequency varies. By contrast, the window pulse method works by scanning a window function (stored in a table) at the same rate regardless of the fundamental frequency, and the frequency is changed by varying the dead space between window pulses. A disadvantage with this method is that the maximum fundamental frequency is limited to about 30% of the half-sample frequency. Another disadvantage is that sound power increases with increasing frequency, but this can be compensated by a frequency-dependent amplitude multiplier.

3.6 Use of Filters in Subtractive Synthesis

Given an input signal, rich in harmonic partials, a filter's task is to shape the input spectrum into something more musically interesting. The "formant" is a concept used in both speech and musical sound analysis, and is roughly synonymous with the concept of resonance: It corresponds to a spectral region of relative emphasis. Acoustical instrument and vocal sounds are often characterized by one or more prominant formants, and filters are powerful devices for simulating these formants. The term "subtractive" comes from the filter's ability to remove frequencies from the input signal or attenuate the amplitudes of frequencies on a selective basis.

The way a filter modifies an input spectrum is best understood through its frequency response characteristic and by understanding the superposition property of linear filters: that a filter's response to the combined sum of two input signals is identical to the sum of the responses of two like filters individually acting upon the signals. The effect of frequency response in shaping an input spectrum is illustrated in Figure 3-8. This can also be summarized by the formula

$$\tilde{c}_{k_{out}} = H(jk2\pi f_1)\,\tilde{c}_{k_{in}} \qquad [3.21]$$

where $H(jk2\pi f_1)$ is the complex frequency response of the filter.

The superposition principal can be stated as

$$\text{Filter}[s_1(t) + s_2(t)] = \text{Filter}[s_1(t)] + \text{Filter}[s_2(t)] \qquad [3.22a]$$

Or if we let h(t) be the impulse response of the filter, we can write this expression as

$$[s_1(t) + s_2(t)]*h(t) = s_1(t)*h(t) + s_2(t)*h(t)\,. \qquad [3.22b]$$

where * signifies convolution.

Note that a great number of devices used in electronic music and audio can be classified as filters. For example, microphones, loudspeakers, reverberators, horns, and rooms are filters, to name a few. All filters are characterized by resonances and anti-resonances (sometimes called poles and zeros), although these effects may not be obvious from the filters's frequency response characteristics.

Equation 3.21 can be extended for the continuous spectrum case. If $S_{in}(j\omega)$ and $S_{out}(j\omega)$ are the Fourier transforms of the filter input and output signals, respectively, they are related by the formula

$$S_{out}(j\omega) = H(j\omega)\,S_{in}(j\omega)\,. \qquad [3.23]$$

Like spectra, filter transfer functions are characterized by their magnitude and phase parts. Minimum phase transfer functions (those whose zeros lie in the left half plane) have an interesting property: The phase function can be derived from the magnitude function through the **Hilbert transform**. This does not hold for all-pass filters (filters whose magnitude is always unity), but it does hold for a great many useful filter transfer functions.

3.6.1 Useful Filter Functions

A family of useful "all-pole" filters can be derived which have the basic form

$$H(s) = 1/(1 + a_1 s + a_2 s^2 + a_3 s^3 + \ldots + a_n s^n) . \quad [3.24]$$

These filters tend to be **Low Pass** since their response goes to zero as the complex frequency s goes to infinity. They can be scaled for any cutoff frequency ω_c by the substitution

$$s \leftarrow s/\omega_c \quad [3.25a]$$

They can be changed to **Band Pass** using the substitution

$$s \leftarrow Q(s/\omega_c + \omega_c/s) \quad [3.25b]$$

They can be changed to **High Pass** using the substitution

$$s \leftarrow \omega_c/s \quad [3.25c]$$

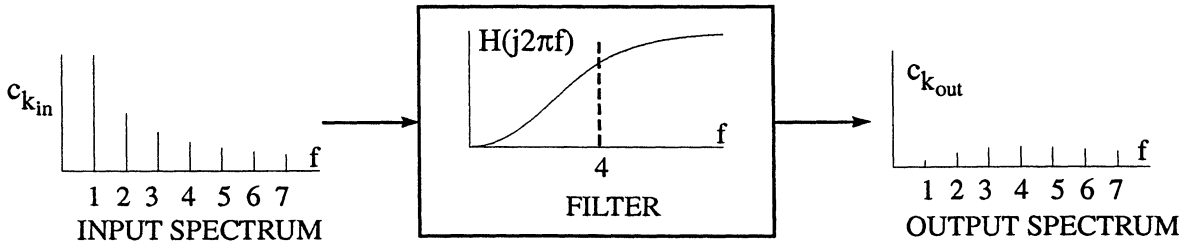


Figure 3-8. Modification of an input spectrum by a filter with a high pass frequency response.

3.6.2 First and Second Order Filters

These lower-order filters can be used as components in higher-order filters, so it is well to understand their operation first. The first order low pass filter is characterized by a gradual cutoff slope and is only useful when very moderate filtering is needed or as part of a more complex filter. Its s-variable response

$$H(s) = 1/(1 + s/\omega_c) \quad [3.26a]$$

can be analyzed for its magnitude and phase components:

$$|H(j\omega)| = 1/\text{sqrt}[1 + (\omega/\omega_c)^2] \quad [3.26b]$$

$$\theta(j\omega) = -\text{atan}(\omega/\omega_c) . \quad [3.26c]$$

The equivalent band pass filter is a second order filter whose response function is obtained by making the substitution of 3.25b into Equation 3.26a:

$$H(s) = 1/[1 + Q(s/\omega_c + \omega_c/s)] \qquad [3.27a]$$

In terms of magnitude and phase this becomes

$$|H(j\omega)| = 1/\text{sqrt}[1 + Q^2 (\omega/\omega_c - \omega_c/\omega)^2] \qquad [3.27b]$$

$$\theta(j\omega) = \text{atan}[Q(\omega_c/\omega - \omega/\omega_c)] \; . \qquad [3.27c]$$

This band pass filter is interesting for its "resonance frequency", $\omega_c = 2\pi f_c$, and its "Q", which gives the degree of selectivity of the filter. The higher the Q, the higher the selectivity, since the - 3 dB bandwidth around the resonance frequency is given by

$$\Delta f = f_c/Q \; . \qquad [3.27d]$$

Thus, this simple band pass filter is a powerful tool for altering spectra and, in particular, for creating "formants" in spectra.

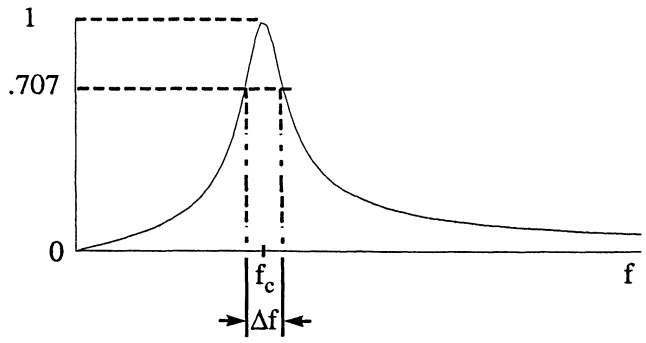


Figure 3-9. Simple band pass resonator frequency response characteristic.

Another second order transfer function, a low pass prototype, is the Butterworth second order filter:

$$H(s) = 1/[1 + \sqrt{2}s + s^2] \qquad [3.28a]$$

The scaled magnitude function for this filter is given by

$$|H(j\omega)| = 1/\text{sqrt}[1 + (\omega/\omega_c)^4] \qquad [3.28b]$$

The response is very flat for $\omega < \omega_c$ and rolls off at - 12 dB/octave above the cutoff frequency.

3.6.3 **Higher Ordered Filters**

Filters with orders higher than two can be constructed to provide very sharp cutoff low pass, band pass, and high pass filters. For example, the Butterworth nth order prototype is a very powerful filter of this

type. It has the property of having a very flat response for $\omega < \omega_c$ and a roll off above cutoff of -6n dB/octave. The nth order magnitude response is given by

$$|H(j\omega)| = 1/\text{sqrt}[1 + (\omega/\omega_c)^{2n}] \qquad [3.29]$$

which has the advantage that it is easy to remember.

Design of filters for sharp cutoff and other classic problems are the subject for a course in filter synthesis. However, these types of filters are not necessarily best for synthesis. For example, a filter often used in analog electronic music has the transfer function

$$H(s) = 1/(1 + s/\omega_c)^4 \qquad [3.30]$$

While this eventually rolls off at -24 dB/octave, it is decidedly "sloppy" at the pass-band/stop-band transition. And yet for producing certain types of musical sounds (assuming a sawtooth input), it is reputed to sound better than a sharp-cutoff 4th order equivalent.

3.6.4 **Realization of Filters**

Filters can be realized using many different technologies. The two principle methods are to use integrators and time delays. These methods work very with analog and digital signals, respectively. Therefore, analog filter design usually makes use of integration devices such as capacitors and inductors, whereas digital filter design normally takes advantage of the inherent time delays afforded by digital memory.

3.6.4.1 **Analog Filters**

Analog filters are generally constructed from resistors, capacitors, inductors, and active electronic devices. A detailed discussion of filter design would be covered in a course in analog network synthesis.

3.6.4.2 **Digital Filters**

Digital filters are designed using time delays (in units of the sample period), adders (subtractors), and gain (or attenuator) units. If $x_0, x_1, x_2, \ldots$ is a series of input samples and $y_1, y_2, y_3, \ldots$ is a series of output samples, the general formula for digital filter is given by

$$y_k = a_0 x_k + a_1 x_{k-1} + a_2 x_{k-2} + \ldots - b_1 y_{k-1} - b_2 y_{k-2} - \ldots \qquad [3.31a]$$

Index k refers to the "current sample", whereas k-1 and k-2 are previous sample values which are delayed by 1 and 2 sample periods ($T = 1/f_s$), respectively. $a_0, a_1, a_2, \ldots$ and $b_1, b_2, b_3, \ldots$ are the filter coefficients, and these constants govern the behavior of the digital filter. Therefore, if we assume a sinusoidal solution

$$x = X e^{-j\omega t}, \; y = Y e^{-j\omega t}, \qquad [3.31b]$$

we can rewrite Equation 3.31a as

$$Y\, e^{-j\omega t} = a_0 X\, e^{-j\omega t} + a_1 X\, e^{-j\omega(t-T)} + a_2 X\, e^{-j\omega(t-2T)} + \ldots - b_1 Y\, e^{-j\omega(t-T)} - b_2 Y \exp^{-j\omega(t-2T)} - \ldots \quad [3.31c]$$

If we eliminate $e^{-j\omega t}$ from all terms and solve for Y/X we have

$$Y/X = H(z) = [a_0 + a_1 e^{-j\omega T} + a_2 e^{-j\omega 2T} + \ldots]/[1 + b_1 e^{-j\omega T} + b2\, e^{-j\omega 2T} + \ldots] \quad [3.31d]$$

In digital filter terminology $e^{-j\omega T}$ is notated as z^{-1}, which stands for a unit sample delay.

By careful control of the a_k's and the b_k's we can achieve almost any desired filter characteristic. Procedures for mapping from the s domain of analog filters to the z^{-1} domain, notably the **bilinear transform** method, are available to simplify the design process. Note, however, that in any case the frequency response Y/X is periodic with frequency domain period f_s.

3.6.4.2.1 Bilinear Transform Digital Filter Design Technique

The Bilinear Transform provides a simple means for deriving a digital filter in the form of H(z) from an analog H(s) function, which preserves the response of the analog filter at f = 0 and another frequency (usually taken to be $f = f_c$) and maps the original response at $f = \infty$ to $f = f_s/2$. For a filter which is defined in terms of a single cutoff or resonance frequency, this usually provides an adequate representation. Since digital filters can not operate above $f = f_s/2$ anyway, it seems reasonable to map the region between a filter's cutoff frequency and infinity to the same cutoff and the half sample frequency.

The transformation is performed by making the substitution

$$s/\omega_c \leftarrow \cot(\pi f_c/f_s)\,(1 - z^{-1})/(1 + z^{-1}) \quad [3.32a]$$

The frequency response mapping can be seen by substituting s by $j\omega$ on the left side and z^{-1} by $e^{-j\omega T}$ on the right side:

$$j\omega/\omega_c \leftarrow \cot(\pi f_c/f_s)\,(1 - e^{-j\omega T})/(1 + e^{-j\omega T}) = j \tan(\pi f/f_s)/\tan(\pi f_c/f_s)\,. \quad [3.32b]$$

This substitution gives the desired mapping properties.

The actual derivation of a recursion formula is easily demonstrated for the simple first-order case: Let $H(s) = 1/(1 + s/\omega_c)$. Then after making the substitution according to Equation 3.32a, we get

$$\begin{aligned} H(z) = Y(z)/X(z) &= 1/(1 + \cot(\pi f_c/f_s)\,(1 - z^{-1})/(1 + z^{-1})) \\ &= (1 + z^{-1})/((1 + z^{-1}) + \cot(\pi f_c/f_s)\,(1 - z^{-1})) \\ &= (1 + z^{-1})/(1 + \cot(\pi f_c/f_s) + (1 - \cot(\pi f_c/f_s))\, z^{-1}) \end{aligned} \quad [3.33a]$$

Let's define $b_0 = 1 + \cot(\pi f_c/f_s)$ and $b_1 = 1 - \cot(\pi f_c/f_s)$. We can then rearrange and write the terms

as follows:

$$Y(z) = (1/b_0)(1 + z^{-1})X(z) - (b_1/b_0)z^{-1}Y(z) \qquad [3.33b]$$

Interpreting Y(z) as the current output sample y_n, $z^{-1}Y(z)$ as the previous output sample y_{n-1}, X(z) as the current input sample x_n, and $z^{-1}X(z)$ as the previous input sample x_{n-1}, we have

$$y_n = (1/b_0)(x_n + x_{n-1}) - (b_1/b_0)y_{n-1} \qquad [3.33c]$$

as the final recursion formula. This is an example of an infinite impulse response (IIR) filter.

Figure 3-10 shows a comparison of an first order analog filter response with the bilinear digital response for the case where $f_c = 0.25\, f_s$. The digital response is obtained by using equation 3.26b with the substitution given by equation 3.32b. In general, the smaller the ratio f_c / f_s is, the less apparent distortion of the response there will be in converting from the analog to the binlinear digital response.

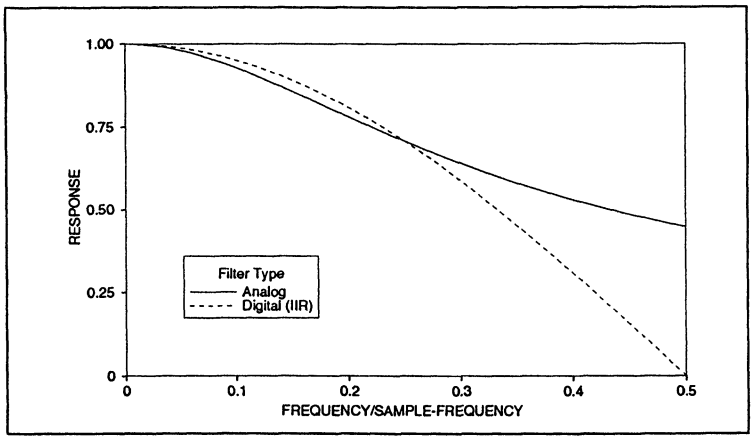


Figure 3-10. Magnitude frequency response of analog and digital first order low pass filters with the same cutoff-frequency/sample-frequency ratio (= 0.25).

3.7 **Envelope Generators**

An **envelope** is a function of time which is used to characterize or control the gradual change of a sound parameter. It generally consists of three segments, a beginning (attack), a middle ("steady-state"), and an end (release). Although it is generally associated with amplitude change, an envelope could also be used to control other parameters, such as frequency. For an isolated tone an amplitude envelope usually starts from zero and ends at zero but may be quite complex in between. However, if a tone is connected to another tone (legato performance), the amplitude will probably not touch zero between the two but may dip to some low value during the transition between tones.

In the simplest case, the amplitude envelope is distinguished by its attack time, length of steady-state, and its release (or decay) time. In the case of percussion sounds the attack time is very short (effectively zero) and the steady-state does not exist; thus a percussion envelope is for all practical purposes simply a decay

envelope. Envelopes useful for mimicking sustained tone instruments (i.e., bowed strings or wind instruments) have attack times which might vary from 5 ms to 500 ms. Release times are generally longer than those of attacks and may vary from 50 ms to .5 sec. (for sustained tone) or up to 20 sec. (for percussion tones, e.g., bells). Linear functions seem to work quite well for attacks, but for decays exponential functions, which yield fixed negative slopes in terms of decibels-per-second (negative), seem much more natural.

The ADSR (attack-decay-sustain-release) envelope was introduced by Robert Moog in the 1960's. This allows for "overshoot" beyond the sustain level during the attack/decay portion of the envelope. The decay does not return to zero (generally) but rather to the sustain level, which is less than or equal to the peak level. The release is the final decay and is associated with the final release of a sustained tone. In the case of keyboard performance the release phase occurs when the key is released. However, the shape of the envelope can be controlled considerably by judicious setting of the sustain level: If set to maximum value, no overshoot occurs (soft attack). If set to zero level, the sound decays to zero after the

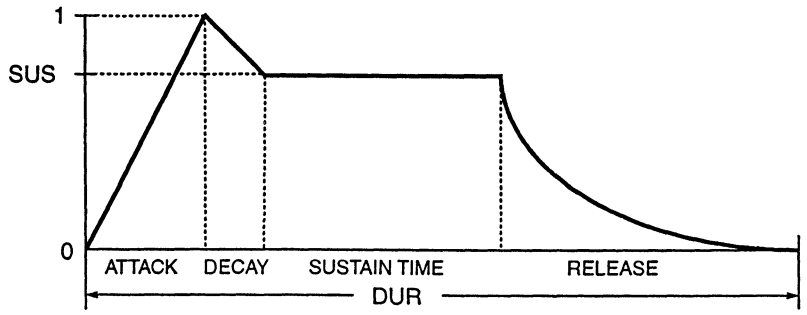


Figure 3-11. An ADSR envelope function.

attack and the release has no effect. If set in between max and zero, overshoot occurs and the result is often a sustained tone with "bite" in the envelope's attack.

Amplitude envelopes which occur in acoustic instrument sounds are actually much more complex. (See Chapter 4 for examples.) They can be approximated by series of straight line or exponential segments, and the number necessary can vary from 8 to 80, depending on the situation. Also, modulatory components (tremolo) can occur which might be better simulated using an LFO (low frequency oscillator) or low frequency noise, in addition to the normal envelope function.

Frequency envelopes are perhaps not as crucial as amplitude envelopes. Nevertheless, the use of frequency deviation functions which bend the frequency slightly (portamento) or produce very rapid variations during the attack can be effective in producing interesting and/or natural-sounding results. The use of low frequency periodic variations (vibrato) is also important for many types of sounds. However, absolutely periodic vibrato is not generally as effective as quasi-periodic vibrato having some random variation of frequency deviation amplitude and rate. Rates between 5 and 10 Hz and deviation amplitudes varying from .5 % to over a semitone in extent are most effective.

References

1. Alles, H. G., "A Portable Digital Sound Synthesis System", **Computer Music J.**, Vol. 1, No. 4, pp 5-9 (1977).

2. Bass, S. C. & Goeddel, T. W., "The Efficient Digital Implementation of Subtractive Music Synthesis", **IEEE Micro**, pp. 24-37, Aug., 1981.

3. Cooper, G. R. & McGillen, C. D., **Methods of Signal and System Analysis**, Holt, Rinehart, and Winston, Inc., Chapters 3 - 5 (1967).

4. Dodge, Charles & Jerse, Thomas A. , **Computer Music Synthesis, Composition, and Performance**, Schirmer Books, pp 78-80, 149-153, 155-178 (1985).

5. Gabel, R. A. & Roberts, R. A. , **Signals and Linear Systems**, Wiley (1973).

6. Grey, John M., "Multidimensional Perceptual Scaling of Musical Timbre", **J. Acoust. Soc. Am.**, Vol. 61, pp. 1270-1277 (1977).

7. Harris, F. J. , "On the Use of Windows for Harmonic Analysis with the Discrete Fourier Transform", **Proc. IEEE**, Vol. 66, pp 51-83 (1978).

8. Helmholtz, Hermann, **On the Sensations of Tone**, Dover Press (1962/1954)

9. Horner, Andrew & Beauchamp, James, "A genetic algorithm based method for synthesis of low peak amplitude signals", **J. Acoust. Soc. Am.**,Vol. 99, No. 1, pp. 433-443 (1996).

10. Moore, J. A. , "The Synthesis of Complex Audio Spectra by Means of Discrete Summation Formula", **J. Audio Engr. Soc.**, Vol. 24, pp 717-727 (1976).

11. Plomp, J. & Steeneken, H.J.M., "Effect of Phase on the Timbre of Complex Tones", **J. Acoust. Soc. Am.**, Vol. 46, pp. 409-421 (1969).

12. Sampson, Peter, "A General-Purpose Digital Synthesizer", **J. Audio Engr. Soc.**, Vol. 28, 106-113 (1980).

13. Schroeder, M.R., "Synthesis of Low-Peak-Factor Signals, ...", **IEEE Trans. Info. Theory**, Vol. IT-16, p. 85 (1975).

14. Wessel, David L., "Timbre Space as a Musical Control Structure" **Computer Music J.**, Vol. 32, No. 2, pp 45-52 (1979).

Time-Variant Spectrum Analysis

CONTENTS

TIME-VARIANT SPECTRUM ANALYSIS

4.0 Introduction

Since acoustic musical instruments (including voice) produce sounds which are complex, time-varying, and nonperiodic, they cannot be modelled as constant waveforms. Complex models are required for realistic synthesis. Therefore, methods of analysis are required which can deal with the complex behavior and produce meaningful parameters to effectively control these synthesis models.

Some types of instruments are easier to analyze than others. We can divide all acoustical instruments into two families: Those which produce harmonic (or approximately harmonic) partials (e.g., a trumpet) and those having decidedly inharmonic partials (e.g., a cymbal). Of the inharmonic spectra there are those whose partials are spaced more closely than harmonic partials and those which are spaced further apart by comparison. From a spectrum analysis point of view, tones with many densely-spaced partials are much more difficult to analyze than those with widely-spaced ones. Harmonic partials represent a particularly easy case since the ordinary method of Fourier series (discrete Fourier transform) can be extended and directly applied for this case.

The output of an analysis method can usually be used as input for synthesis. Once a method of analysis and complementary synthesis is realized, it is possible to think of this combination as a method for **sound processing**. For example, since Fourier series analysis yields amplitudes and frequencies, it is possible to scale these to shift pitch and modify spectrum. Duration can also be scaled without affecting pitch and spectrum.

4.1 Sound Analysis: A Generalized Fourier Series Approach

This method works best for signals which are quasi-periodic, i.e., signals for which

$$s(t + T) \cong s(t). \qquad [4.1]$$

In particular, we can think of a quasi-periodic signal as one which has a definite period but differs from absolute periodicity in one or more of the following ways:

1) Amplitude is changing with time (particularly during the attack and decay of a tone).

2) Waveform is changing with time (again, especially during the transients).

3) Period and average fundamental frequency vary around their average values, T and f_a.

4) Phase (or frequency) of each harmonic varies around its mean value independently of the other harmonics.

5) Amplitudes of the various harmonics vary independently of one another.

6) Mean frequencies of the partials are not strictly related as integer multiples. For example, in

struck/plucked string tones, the partial frequencies become increasingly sharp compared to the harmonic ideal as the harmonic number increases.

Despite the possibility that the signal many have a time-varying period, the method described here uses a constant period for analysis, close to the inverse of the signal's average fundamental frequency. Thus,

$$T = 1/f_a \qquad [4.2]$$

where f_a is the analysis frequency, close to the average frequency of the input.

One way to think about time-variant analysis is in terms of a "stepped-window" or "sliding window" analysis. For each successive "period" of the signal a Fourier series analysis is performed. The time-variant analysis then becomes the collection of ordinary Fourier series analyses for successive periods or "windows". It is a theorem of Fourier analysis that this series will converge at every point within each period -- although not at the end points if the beginning and end of the period do not match. The latter problem is easily overcome, however, if we analyze twice as many periods and overlap by a half-period. Since the series will surely converge on the middle half of each period analyzed, all points can be made to converge. Alternatively, if the signal is bandlimited, a finite number of harmonics will converge over the entire period. Thus, the analysis technique is inherently **complete**. A general formulation of the time-variant Fourier analysis, using complex exponentials is as follows:

$$\tilde{c}_k = \frac{1}{T}\int_{t-T/2}^{t+T/2} s(x)\, e^{-j\omega_k x}\, dx = \int_{-\infty}^{\infty} w(t-x)\, s(x)\, e^{-j\omega_k x}\, dx \qquad [4.3]$$

where $\omega_k = 2\pi k/T = 2\pi k f_a$ is the kth harmonic radian frequency of analysis. Note that $w(t) = (1/T)\,\mathrm{rec}(t;T)$ is the rectangular window function.

If the signal were sampled into 2n evenly-spaced points per period, a discrete Fourier transform could be used to compute the harmonic components, resulting in exactly n harmonics. Assuming the signal to be bandlimited and sampled according to the Nyquist criterion, there would be no loss of information. The discrete approach would result in sums rather than integrals in the formulas for $c_k(t)$, and t would take on a set of discrete values. In fact, this is usually how time-variant analysis is accomplished in practice, either through real-time digital circuits or by software simulation. However, we will proceed with continuous signals and integrals. Keep in mind that the results arrived at here have very similar counterparts when discrete forms are used.

From Equation 4.3 we can rewrite the complex harmonic amplitude as

$$\tilde{c}_k(t) = w(t) * [s(t)\, e^{-j\omega_k t}], \qquad [4.4]$$

where $*$ represents convolution.

Equation 4.4 illustrates that the kth harmonic component can be obtained by first multiplying by a

complex sinusoid at the frequency of the harmonic and then convolving with a window function. Since the window function acts as a low pass filter, the method is sometimes called the **heterodyne/filter** method of analysis.

The rectangular window function w(t) may be considered to be an impulse response function. The corresponding frequency response is given by its Fourier transform W(jω) which is given by

$$W(j\omega) = \text{sinc}(\omega T/2) = \text{sinc}(\pi f/f_a) = \sin(\pi f/f_a)/(\pi f/f_a) \qquad [4.5]$$

This low pass response is unity at zero frequency and is zero at frequencies which are integral multiples of f1, the assumed analysis frequency. (The rectangular window and its transform were shown in Figure 3-6 a),b),c). of Chapter 3.)

Once the time-variant harmonic amplitudes $\tilde{c}_k(t)$ have been computed, the signal can be synthesized using

$$\hat{s}(t) = \sum_{k=-K}^{K} \tilde{c}_k(t)\, e^{j2\pi k f_a t} \qquad [4.6]$$

where K is a suitably large number. This is additive synthesis with complex harmonic amplitudes. It can be converted to the following real form:

$$\hat{s}(t) = \sum_{k=0}^{K} c_k(t) \cos(2\pi k f_a t + \theta_k(t)) \qquad [4.7]$$

where $c_k(t) = 2\,|\tilde{c}_k(t)|$ and $\theta_k(t) = \arg[\tilde{c}_k(t)]$.

The heterodyne/filter analysis method can be understood in terms of spectral manipulations in the frequency domain. Let's suppose the signal to be analyzed consists of steady-state components with frequencies f_1, $2f_1$, $3f_1$, . . . If we wish to analyze the 4th harmonic, we first multiply the signal by $e^{-j2\pi f_a t}$, where $f_a \cong f_1$. In the frequency domain this has the effect of shifting the spectrum to the left, so that the 4th harmonic component lies at the origin. Figure 4.1a shows a hypothetical input spectrum, and Figure 4.1b shows the version shifted to the left, representing the effect of heterodyning by $4f_a$. For simplicity, we will assume that $f_a = f_1$. exactly, i. e., *there is perfect tuning between the analyzer and the input signal.* Then, if the sinc filter function is applied (Figure 4.1b), it zeros all harmonic frequencies, leaving the c_4 component at $f = 0$ as shown in Figure 4.1c. Thus, this traditional analyzer works perfectly for steady-state periodi (harmonic) signals whose fundamental frequency f_1 perfectly matches the analysis frequency.

Figure 4.2 shows a block diagram depicting the heterodyne/filter analyzer and the subsequent (additive) synthesis stage. With the rectangular windo, this analyzer/synthesizer is complete in the sense that (at least theoretically) the output signal is exactly equal to the input signal. Unfortunately, we can not guarantee that the $c_k(t)$ envelopes are meaningful, smooth functions, whose data can be approximated by smooth curves. "Errors" are caused if, unlike the periodic signal depicted in Figure 4.1, the input signal is not periodic or/and the analysis frequency is different than the frequency of the input signal. In the latter

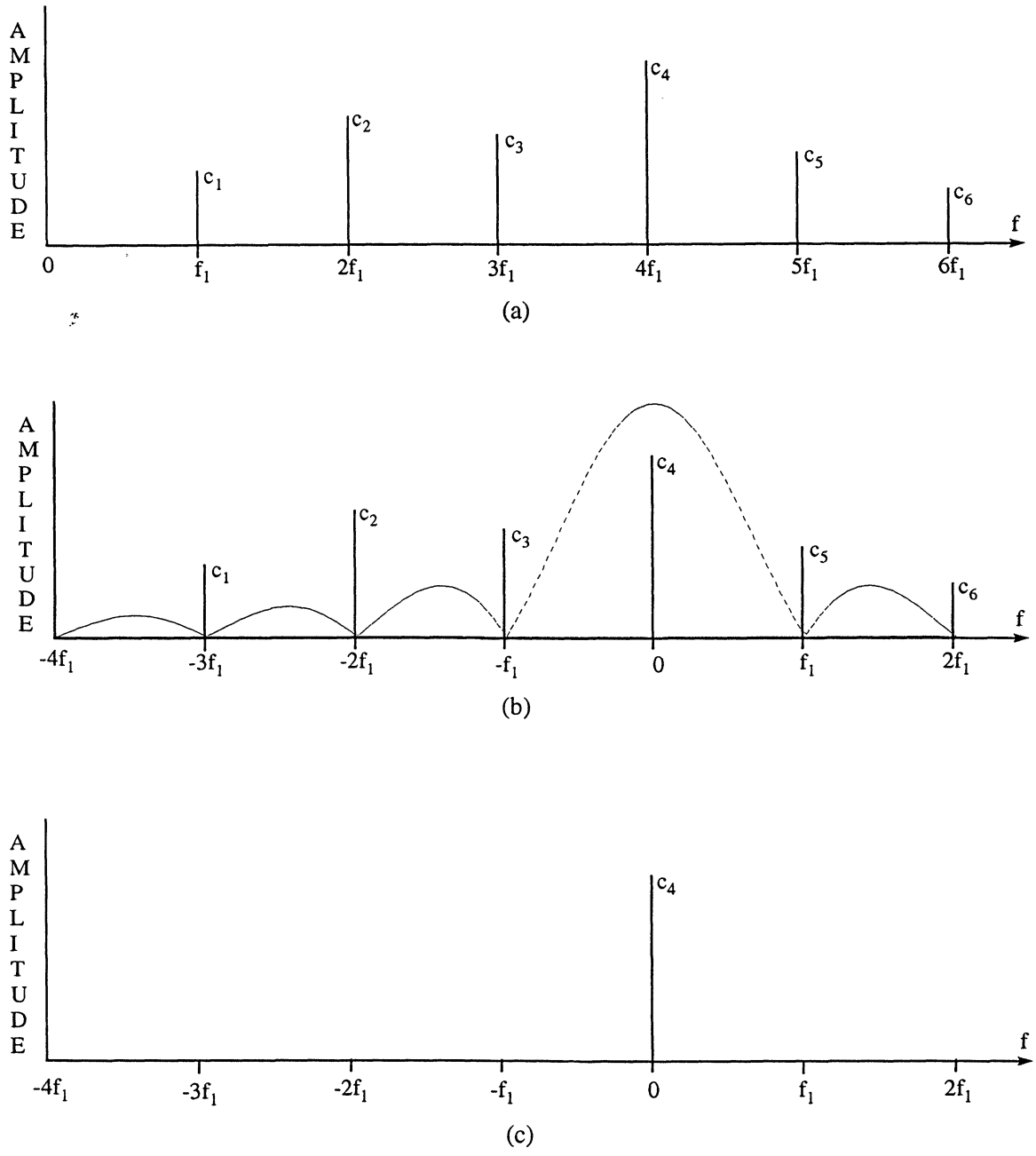


Figure 4.1 Heterodyne/Filter Analysis Process. a) Hypothetical input spectrum. b) Spectrum after heterodyne with frequency $4f_1$ and superimposed sinc low pass filter response (for the case $f_a = f_1$ exactly). c) Result after filter operation.

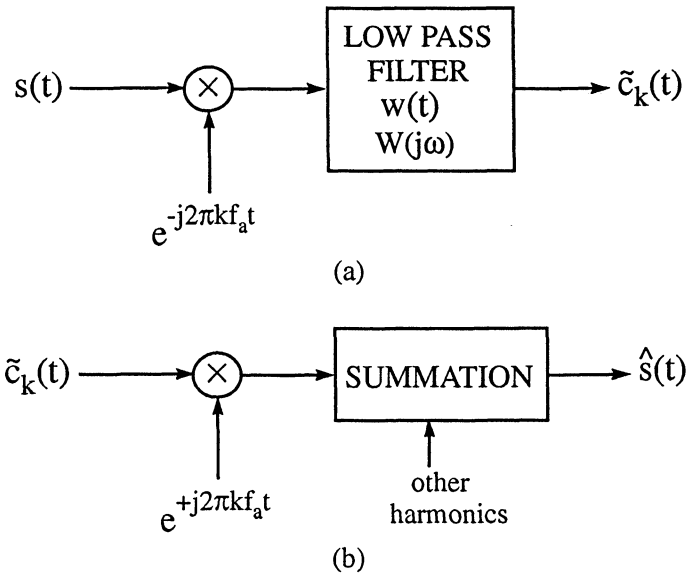


Figure 4.2 a) One channel of a heterodyne/filter analyzer. b) Complementary additive synthesizer block diagram.

case, if the difference between the signal's fundamental and the analysis frequency is small (less than 1% or so), the analysis still works quite well, and the slope of the detected phase of the fundamental can be used to estimate the frequency error. In the former case, where transients occur, the input signals's transform no longer consists of pure lines, but rather components of appreciable width. Therefore, the "harmonic components" are not nulled by the zeros of the sinc function, and, in the time domain, they contribute an error to the detected harmonic amplitude function. Moreover, the time variation of the recovered harmonic may be distorted by the nonuniform response of the sinc function below frequency f_1.

Figure 4.3 shows an example of an idealized first harmonic phase plotted as a function of time for an analysis frequency error of $\Delta f = f_1 - f_a = 2$ Hz. The slope of this sawtooth-like curve divided by 2π gives an estimate of the frequency error (or deviation), which can be used to correct the analysis frequency on a second analysis pass.

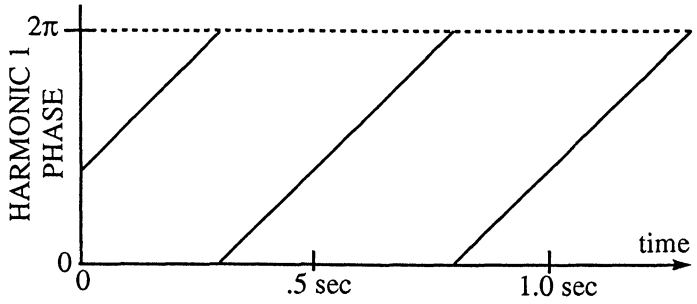


Figure 4.3 Measured harmonic 1 phase. The slope of this curve indicates that the analysis frequency f_a and the input signal's fundamental frequency differ by 2 Hz.

That this would be the result can be seen from the following analysis:

Let $s(t) = \cos(2\pi f_1 t)$. Then

$$\tilde{c}_1(t) = \frac{1}{T}\int_{t-T/2}^{t+T/2} \cos(2\pi f_1 t)\, e^{-j2\pi f_a t}\, dt$$

$$= \frac{1}{2T}\int_{t-T/2}^{t+T/2} e^{j2\pi(f_1 - f_a)t}\, dt + \frac{1}{2T}\int_{t-T/2}^{t+T/2} e^{j2\pi(f_1 + f_a)t}\, dt$$

$$= \text{sinc}(\pi(f_1 - f_a)T)\, e^{j2\pi(f_1 - f_a)t} - \text{sinc}(\pi(f_1 + f_a)T)\, e^{j2\pi(f_1 + f_a)t}$$

$$\approx 1 \bullet e^{j2\pi(f_1 - f_a)t} - 0 \bullet e^{j2\pi(f_1 + f_a)t}$$

The phase of c_1 is $2\pi(f_1 - f_a)t$, which varies with time similarly to the graph of Figure 4.3. The frequency error, or deviation, is just $f_1 - f_a$. Since the second term of this result is not actually zero, a small amount of oscillatory "ripple" would normally appear on the graph.

In general, we can define the **frequency deviation** of the kth harmonic as

$$\Delta f_k(t) = (1/2\pi)\,(d/dt)\,\arg(c_k(t)) \qquad [4.8a]$$

Then, the **instantaneous frequency** of the kth harmonic can be defined as

$$f_k(t) = k f_a + \Delta f_k(t) \qquad [4.8b]$$

"Ripple" can occur from the analysis of a transient signal. Let us demonstrate this with a specific case. Suppose we have a tone consisting of a sum of harmonics based on the fundamental $f_1 = 1/T$, with each harmonic having a different initial amplitude A_i and exponential decay rate $-\alpha_i$:

$$s(t) = \sum_{i=1}^{n} A_i\, e^{-\alpha_i t} \cos(2\pi i f_1 t) \qquad [4.9]$$

Assuming that the analyzer and signal are perfectly in tune ($f_a = f_1$), then the complex Fourier amplitude of the kth harmonic is given by

$$\tilde{c}_k(t) = \frac{1}{T}\int_{t-T/2}^{t+T/2} \left[\sum_{i=1}^{n} A_i\, e^{-\alpha_i x} \cos(2\pi i f_1 x)\right] e^{-j2\pi k f_1 x}\, dx \qquad [4.10a]$$

After a fair amount of manipulation, this can be reduced to

$$\tilde{c}_k(t) = .5\sum_{i=1}^{n} A_i\,(-1)^{k+i}\sinh(\alpha_i T/2)\,e^{-\alpha_i t}\times\left(\frac{e^{-j(\omega_k+\omega_i)t}}{\alpha_i T/2 + j\pi(k+i)} + \frac{e^{-j(\omega_k-\omega_i)t}}{\alpha_i T/2 + j\pi(k-i)}\right) \qquad [4.10b]$$

$$= .5\,A_k\frac{\sinh(\alpha_k T/2)}{\alpha_k T/2}e^{-\alpha_k t} + \text{higher frequency terms.} \qquad [4.10c]$$

As the products of the decay rates and the period approach zero, we see that the complex harmonic amplitude approaches the ideal function, i.e.,

$$\tilde{c}_k(t) \rightarrow .5\,A_k\,e^{-\alpha_k t} \text{ as } \alpha_k T \rightarrow 0\ . \qquad [4.10d]$$

The real $c_k(t)$ is given by $2\,|\,\tilde{c}_k(t)\,|$, and so

$$c_k(t) \rightarrow A_k\,e^{-\alpha_k t}\ , \text{ the result we would expect.} \qquad [4.10e]$$

It is difficult to write a complete equation for the time-dependent harmonic amplitudes for this case, as this involves taking the magnitude of many complex terms. However, the complete amplitude functions can be plotted using complex variable programs (e.g., written in MatLab). The results of analysis for the case where the original signal is given by

$$s(t) = e^{-100t}\cos(2\pi 400t) + e^{-200t}\cos(4\pi 400t) \qquad [4.11]$$

are shown in Figure 4.4 a),b); showing how $c_1(t)$ and $c_2(t)$ approximate the original envelopes and the introduction of an error in terms of added ripple.

The ripple may be thought of as the sum of signals from harmonic components which are not completely removed by the sinc low pass filter. This error may also be thought of as a "crosstalk" or "leakage" between analysis channels. The problem of ripple can be reduced considerably by using window filters with better stop band attenuation than that afforded by the sinc filter (the rectangular window). The hanning, Hamming, and Blackman-Harris window filters can be profitably used since each of these has the very desireable dual property of being zero at all harmonic frequencies and having improved stop band attenation. A tradeoff between ripple rejection and response to rapid transients exists, however, since the hanning and Hamming filters require two periods of integration (the Hamming gives -42 dB rejection) while the B-H filter requires a three or four period integration (it gives a -92 dB rejection). Sometimes "time smearing" of transient details result from using larger window sizes. Even so, *completeness* -- the property that the reconstructed signal is identical to the original -- can be guaranteed if all the harmonics associated with the double (or larger) period are utilized, since the frequency responses of these filters (in their bandpass configuration as defined by Equation 4.14) add to unity. For slowly-changing quasi-harmonic input signals, we expect that only the harmonics of the original period will be needed. For this case, subharmonic terms resulting from the multiple period analysis can usually be safely discarded.

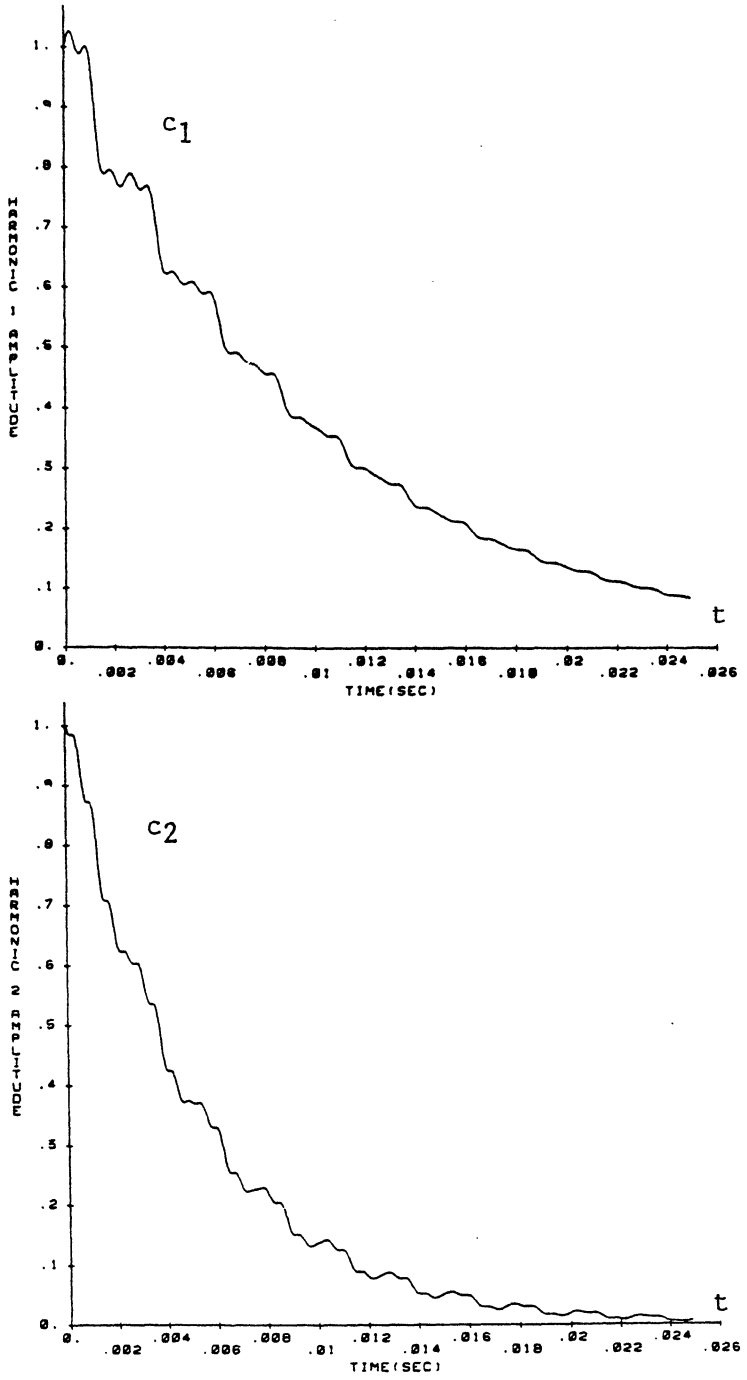


Figure 4.4 Output of the heterodyne/filter analyzer with rectangular window (sync) low pass filter for the first two harmonic amplitudes resulting from the input signal given by Equation 4.11 in the text.

4.2 Another Interpretation: The Phase Vocoder

The analysis/synthesis process can be simplified if we combine Equations 4.3 and 4.6 into a single equation:

$$\hat{s}(t) = \sum_{k=-n}^{n} e^{j\omega_k t} \int_{-\infty}^{\infty} w(t - x)\, s(x)\, e^{-j\omega_k x}\, dx \qquad [4.12a]$$

$$= \sum_{k=-n}^{n} \int_{-\infty}^{\infty} s(x)\, w(t - x)\, e^{j\omega_k (t - x)}\, dx \qquad [4.12b]$$

$$= \sum_{k=-n}^{n} s_k(t) = \sum_{k=-n}^{n} s(t) * [w(t)\, e^{j\omega_k t}], \qquad [4.12c]$$

where, again, * represents convolution. The term inside the summation of Equation 4.12c resembles Equation 4.4 but is actually quite different. In this case, the output signal is obtained by adding a number of components each of which is the convolution of the input signal and a window which has been heterodyned by one of the harmonic frequencies, i.e., a band pass filter centered at a harmonic frequency. This is equivalent to passing the signal through a bank of contiguous band pass filters, one at each harmonic position and then summing the result. This process becomes more obvious if we look at the result of taking the Fourier transform of both sides of Equation 4.12c:

$$\hat{S}(j\omega) = \sum_{k=-n}^{n} S_k(j\omega) = \sum_{k=-n}^{n} S(j\omega)\, W(j(\omega - \omega_k))$$

$$= S(j\omega) \sum_{k=-n}^{n} W(j(\omega - \omega_k)) \qquad [4.13]$$

Since $W(j\omega)$ is a low pass filter, the shifted version, $W(j(\omega - \omega_k))$ must be a band pass filter with center frequency ω_k. Under certain conditions the phase vocoder will be an *identity operation*, i.e., the analysis will be *complete*. It is clear from Equation 4.13 that if we wish to have $\hat{S}(j\omega) \equiv S(j\omega)$, a necessary and sufficient condition is that

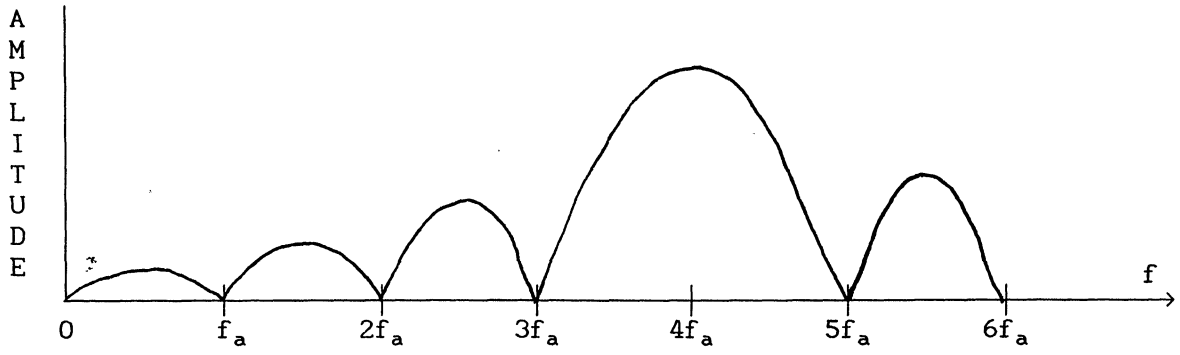
$$\sum_{k=-n}^{n} W(j(\omega - \omega_k)) = 1 . \qquad [4.14]$$

This is tantamount to saying that the sum of the band pass filter responses must be unity, an obvious truth for rectangular (box car) filters. Since we

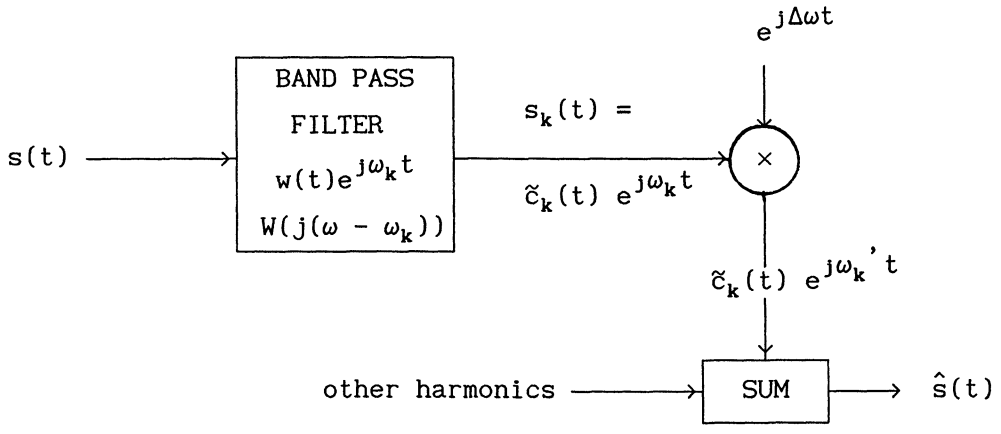
know ordinary Fourier series analysis is complete, it follows that the harmonic set of sinc filters also adds to unity. Not so obviously, the Hamming and Blackman-Harris window responses add to unity if we include all of the harmonics based on the window sizes appropriate for analysis.

Figure 4.5a shows a shifted sinc filter (resulting from a rectangular window response multiplied by a complex sinusoid). Figure 4.5b shows an input-output block diagram suitable for processing a single harmonic component according to the analysis/synthesis method defined by Equation 4.11c, with a modification for shifting the frequency before resynthesis. The reader might compare this with Figure 4.2, which depicts the heterodyne/filter method. Such a system is known as a **phase vocoder**, after the original speech vocoders (voice coders) consisting of banks of band pass filters and various devices for speech data compression and resynthesis. However, the method described here is mathematically equivalent to the heterodyne/low-pass-filter system discussed in Section 4.1.

The phase vocoder method is complete in the sense that it yields an identical copy of the original as long as there is no modification of parameters (i.e., harmonic amplitudes and frequencies) between the analysis and synthesis stages and no information is discarded. The method can be used to process non-harmonic (e.g., speech and polyphonic music) as well as quasi-harmonic sounds to shift their pitches and scale their durations. As one might imagine, this technique works much better if windows such as the Hamming or Blackman-Harris are used, rather than the rectangular window. In the case of nearly harmonic signals where the analysis frequency is set close to the signal's fundamental frequency, these windows exhibit greatly improved crosstalk rejection, resulting in more realistic, smoother parametriccurves. In the case of patently non-harmonic signals, there is no "best" analysis frequency, but good results are still possible, albeit not guaranteed. (Unexpected artifacts can result if the analysis frequency f_a is not chosen carefully.) In either case, the extracted harmonic amplitudes and frequencies will have bandwidths well-confined to the analysis frequency and so may be represented in terms of two values per period, a considerable saving of memory for a digital system. Dolson has demonstrated these possibilities using a computer to process speech and musical sounds [Dolson, 1984, 1985]. Even so, if more than one frequency component exists within a particular bandpass filter channel, the resulting amplitude or frequency data will probably be difficult to interpret and further data reduction will be impossible. Thus, despite this method's general capabilities as a sound processor, it is not very suitable as an analyzer of grossly non-harmonic sounds.



(a)



(b)

Figure 4.5 Phase Vocoder Analyzer/Synthesizer. a) Shifted low pass sinc filter function becomes a band pass filter centered at the harmonic frequency of interest. b) Phase vocoder block diagram (one channel) with frequency shift synthesis capability.

4.2.1 Digital Implementation of the Heterodyne Filter/Phase Vocoder Method

The starting place for analysis is Equation 4.3, which can be also stated in terms of the Fourier transform:

$$\tilde{c}_k(t) = \int_{-\infty}^{\infty} w(t-x)\, s(x)\, e^{-j\omega_k x} dx = \mathcal{F}_{\omega_k x}\{w(t-x)s(x)\} \qquad [4.15]$$

The subscript of the Fourier transform operator is meant to indicate that the integration variable is x and that the Fourier transform is being evaluated at frequency ω_k. In words, this means that to perform this operation we can first multiply the signal s(x) by a window function w(t-x), which focuses on a region around x = t (the signal is said to be *windowed*), take the Fourier transform of the result, and then evaluate the transform at frequency ω_k. To repeat, the computational steps are: 1) choose a starting value of t; 2) multiply w(t-x) by s(x); 3) take the Fourier transform; 4) evaluate at successive values of ω_k; 5) increment t until the required values are exhausted. However, it is obvious that if we only need a finite set of frequencies ω_k, and we are going to evaluate Equation 4.15 by numerical, as opposed to analytical, integration, we might as well only integrate Equation 4.15 at successive discrete values of ω_k in the first place. Thus, steps 3) and 4) can be replaced by "take the Fourier transform at succesive values of frequency ω_k".

For a digital implementation the Fourier integral must be replaced by a discrete approximation formula, which means that the integral's argument must be sampled at discrete points. It also helps if the limits of the integral are finite. Fortunately, both of these issues are easily resolved, since, according to the sampling theorem, band-limited signals can be represented by samples spaced by 1/(2xhighest_signal_frequency) and the window functions we will use are of finite extent. Therefore, if we replace the signal by its sampled version, Equation 4.15 can be rewritten

$$\tilde{c}_k(mT) = \sum_{n=m-N/2}^{m+N/2-1} w(mT - nT)\, s(nT)\, e^{-j\omega_k nT}(1/N) \qquad [4.16a]$$

where N is the length of the window function w() in samples. Equation 4.16a would be sufficient if we wanted to simply compute the time-variant transform at an arbitrary set of frequencies, ω_k. We can just do a straight forward multiplication and summation to calculate the complex partial. However, if we particularize this equation to a set of harmonics based on the length of the window we obtain

$$\tilde{c}_k(mT) = \frac{1}{N} \sum_{n=m-N/2}^{m+N/2-1} w((m - n)T)\, s(nT)\, e^{-j2\pi kn/N} \qquad [4.16b]$$

This is almost in the form of the well-known **Discrete Fourier Transform (DFT)**.

To move it into this form, the summation should start from zero. To accomplish this, let us define n = n'+m-N/2, substitute this into 4.16b, and then drop the primes:

$$\tilde{c}_k(m\tau) = e^{j\pi k(1-2m/N)} \frac{1}{N} \sum_{n=0}^{N-1} w((N/2-n)\tau)\, s((n+m-N/2)\tau)\, e^{-j2\pi kn/N} \quad [4.16c]$$

The summation part of Equation 4.16c is a DFT, although we should note that each kth term of the DFT result is multiplied by a complex exponential, which has the effect of shifting the term's phase. Also, since the summation is a DFT, we can compute it using a **Fast Fourier Transform** (FFT) algorithm.

We also need to consider what the spacing for values of m should be. This is referred to as the *hop size*. The greater the hop size, the less data that needs to be stored in a digital system. It turns out that the hop size must be .5/(BW•τ) or less, where BW is the bandwidth of the window function. For the rectangular window, the bandwidth is very large, so the only really appropriate hop size is h = 1 sample. (In practice, we can often get away with a greater value.) For the hanning or Hamming window, the hop size should be h = .25N, and for the Blackman-Harris it should be h = .125N. This has implications for the exponential multiplier of Equation 4.16c.

In words, the procedure for computing the complex coefficient $\tilde{c}_k(m\tau)$ is as follows:

A) Sample the window function w(*n*τ) from -Nτ/2 to Nτ/2-1 and put values into an array indexed from 0 to N-1, where w(0) is at index N/2. This array will be fixed for the duration of the analysis.

B) For each time value mτ (starting from 0, incremented by hτ):

1) Sample the signal over the interval (m-N/2)τ to (m+N/2-1)τ and put values into an array indexed from 0 to N-1, where s(mτ) is at index N/2. The N samples should correspond to some multiple of the signal's assumed fundamental period, where the multiple depends on the type of window being used. It might be better to say "resample the signal", as the signal's sample rate is probably not appropriate to give the correct value of N for the FFT operation. Most efficient FFT algorithms require N to be a power of 2. To resample the signal, a band-limited interpolation algorithm is needed [Smith and Gossett, 1984].
2) Multiply the corresponding values of the arrays of steps 1) and 2) and put values into an array of complex values indexed from 0 to N-1. Note that only the real parts of the complex values will be non-zero.
3) Take the Fast Fourier Transform (FFT) of the complex array of step 3). The result will be a complex array of size N. After multiplying all array values by $e^{j\pi k(1-2m/N)}$, the array will contain the $\tilde{c}_k(m\tau)$ complex coefficients indexed from k=0 (DC), k=1 (first harmonic), to k=N/2-1 (the (N/2-1)th harmonic. The remaining values in the array (indexed N/2 to N-1) have the same magnitude and phase information as the $\tilde{c}_k(m\tau)$ values for k=N/2 to 1 (i.e., the harmonics in reverse order) and thus represent

redundant information and can be ignored.

Simplifications may be in order for particular window functions. For example, if the Hamming window is used, we can take m = ph = .25pN, where p = 0, 1, 2, ... In this case, the complex exponential multiplier referred to above reduces to $e^{j\pi k(1-.5p)}$, which, in effect, means phase shifts by multiples of 90°. Also, when the Hamming window is used, we would set N to correspond to two periods of the signal's fundamental. Thus, the "harmonics" which result actually correspond to integer multiples of .5 f_1. Thus, if we are only interested in harmonics of f_1, we can chose new harmonics k' = k/2, and discard $\tilde{c}_k$ for odd values of k. The complex exponential multiplier will now cause phase shifts which are either 0° or 180°. A similar result would obtain for the Blackman-Harris window, where we would take m = .125p, and could keep $\tilde{c}_k$ for only k = 4, 8, 12, ...

4) Amplitudes and phases of the harmonics are now calculated. They are given by

$$c_k(mT) = 2\left|\tilde{c}_k(mT)\right| = \sqrt{a_k(m)^2 + b_k(m)^2} \qquad [4.17a]$$

$$\theta_k(mT) = \arg[\tilde{c}_k(mT)] = \operatorname{atan}\{b_k(m)/a_k(m)\} \qquad [4.17b]$$

where $a_k(m)$ and $b_k(m)$ are twice the real and imaginary parts of $\tilde{c}_k(mT)$, respectively.

5) Only the initial values of θ_k need to be kept if we are going to calculate the frequency deviations of the harmonics. The frequency deviations can be calculated using

$$\Delta f_k(mT) = \frac{1}{2\pi hT}\{\theta_k(mT) - \theta_k((m-h)T\} = \frac{1}{2\pi hT}\Delta\theta_k(mT)$$

$$= \frac{1}{2\pi hT}\operatorname{atan}\left(\frac{b_k(n)a_k(n-h) - a_k(n)b_k(n-h)}{a_k(n)a_k(n-h) + b_k(n)b_k(n-h)}\right) \qquad [4.17c]$$

The total frequency of each harmonic is then

$$f_k(mT) = k\,f_1 + \Delta f_k(mT) \qquad [4.17d]$$

In summary, we have described a digital technique for computing the time-variant amplitudes and frequencies of harmonics of a complex quasi-periodic signal. This data can then be modified and used for resynthesis using a procedure indicated by Equation 4.7 or 4.18(additive synthesis).

4.3 The McAulay-Quatieri Time-Variant Spectrum Analysis/Synthesis Method

The time-variant method described so far has a number of limitations. First, for analysis to work well, each sine wave in the signal should be confined to a single frequency channel and there should only be one sine wave in each channel. Usually, this means that the method is restricted to the analysis of quasi-periodic sounds, i.e., sounds with harmonic partials and nearly constant pitch. Also, the base frequency of the analyzer should be close to the fundamental frequency of the sound under analysis. Thus, the pitch of the signal needs to be known prior to the analysis. Also, sounds with large frequency swings, grossly inharmonic sounds, and multiphonic sounds are excluded. An analysis technique developed by McAulay and Quatieri [1984, 1986], to a large extent, circumvents all of these limitations. Their method was originally designed for efficient coding of speech sounds. Smith and Serra [1987, 1989] have further enhanced this technque for analysis/synthesis of musical sounds, and Maher [1991] has used the method for separation of voices in duet performances.

Like the analyzer previously described, the McAulay-Quatieri (MQ) technique uses an arithmetically-spaced, fixed-frequency filter bank. This is inherent in using a Fast Fourier Transform. In contrast, however, the base frequency of the MQ method is not adjusted to match the signal's fundamental; instead it is chosen to be a fixed frequency, independent but substantially lower than the fundamental of the incoming signal. In the method described in Sectionas 4.1 and 4.2, amplitude and frequency were determined directly from individual filter outputs. With the MQ method, amplitude, frequency, and phase are calculated from estimated *peaks* of the Fourier magnitude spectrum at each time instant. The peaks are *tracked* from one time instant to the next, and only peaks which exist for a certain amount of time and are above a certain amplitude threshold are considered to be valid. The result is a series of individual sinusoids (also called *peaks* or *tracks*) with variable frequencies, phases, and amplitudes which can be summed to form the reconstituted output signal:

$$\hat{s}(t) = \sum_{k=0}^{n} c_k(t) \cos(2\pi \int f_k(t)dt + \theta_k(t)) \qquad [4.18]$$

Theoretically at least, it should be possible to absorb variable phase terms into corresponding variable frequency terms, except for starting phase values. For many kinds of sounds, the phase information can be simply eliminated without much loss of fidelity. Note that the partials are not necessarily harmonic, because the frequencies, f_k, are arbitrary. Also, not all partials are "on" all the time; the technique allows partials to die and be born at various times throughout the sound as they rise above and sink below the arbitrary amplitude threshold.

The MQ technique opens up the possibility for analysis of inharmonic sounds, sounds with widely varying frequencies, and polyphonic sounds. For a harmonic sound, it is not necessary to know a signal's fundamental frequency in advance, since all frequencies are located by estimating spectrum peaks. Moreover, the spectrum data can be used to estimate the fundamental frequency

of a harmonic signal. Separation of voices by separating groups of harmonics can be attempted, and this works fairly well [Maher, 1990]; however, "collisions" of partials, which occur whenever individual frequencies belonging to the original two or more sounds lie within a common analyzer frequency band , makes it difficult to separate some partials. Serra and Smith [1990] have demonstrated separation of statistical noise from predictable sine wave content using a variation on this technique. This yields a dramatic improvement in controlling the synthesis of sounds which have a significant amount of noise content.

Ordinarily the MQ technique does not provide a series of harmonically related partials, but it can be coaxed to do so. To analyze a quasi-periodic sound in terms of harmonic partials with the MQ technique requires three steps: 1) identification of constituent sine wave components; 2) determination of the fundamental frequency (varying with time) from the set of frequencies available; 3) reduction of sine wave frequencies to a harmonic set. This is a fairly involved procedure compared to the straightforward heterodyne filter/phase vocoder approach, and perhaps not as robust, when the approximate fundamental is known in advance and the frequency variations are slight. In conclusion, it seems that the HF/PV approach should be used whenever it seems applicable, but if the special capabilities of the MQ technique are needed, it may become indispensible.

4.4 Techniques for Frequency (Pitch) Extraction

Fundamental frequency or pitch estimation is a non-trivial problem in general since musical sounds are not always "well-behaved". Problems result when the fundamental and other lower harmonics are weak compared to higher partials, when their is substantial noise content or inharmonicity, when reverberation is present, and when notes are performed at different pitches in rapid succession. Pitch estimation is part of a larger problem of automatic music transcription from acoustic input. A few attempts have been made to solve this problem, but no robust solution has been published. Part of the problem may be in how we hear pitch (subjective) rather than relying on a simple objective measure, such as the determination of a repetitive period or lowest frequency. Nevertheless, we will look at methods which rely on objective measures of period or frequency.

Note that we will assume that the input signal is due to a monophonic source. The problem of detecting pitch in the presence of other interfering sources (the polyphonic case) is much tougher and beyond our scope here.

Pitch estimation methods divide into two types: 1) time-domain signal-based methods; 2) frequency-domain spectrum-based methods.

4.4.1 Time Domain Methods

The simplest method is to count zero crossings or measure the time between zero crossings. Another possibility is to apply the same measures to the peaks of waveforms. These are expedient methods, but they are easily fooled since waveforms often have more than two zero crossings and more than one peak per period. It is possible to improve the performance of this type of pitch detector by using nonlinear distorters and low pass filters to enhance the fundamental.

The best method of this type seems to be the **subtractive correlation** method [Moorer, 1974; Ross et al , 1974]. A copy of the signal is successively shifted by time τ and subtracted from itself over a fixed interval T. The result is integrated. If the signal is absolutely periodic, the integral will be zero when τ equals the period of the signal. In a real case, the integral will be minimum not zero. This may sound fool-proof, but it turns out that there are usually several minima in addition to the correct one. Hopefully, the correct one will be the lowest of all. The trick is to identify the correct minimum. At any rate, the idea is to find the value of τ which minimizes

$$\int_{t-T/2}^{t+T/2} |s(x+\tau/2) - s(x-\tau/2)|d\tau \qquad [4.19]$$

At time t the detected pitch is then measured as $\hat{f}_1 = 1/\tau$. This operation is repeated for successive values of t so that a pitch vs. time function $\hat{f}_1(t)$ is recovered from the signal. This could then be translated into musical pitch vs. time using equations from Chapter 2 (Section 2.1.1).

4.4.2 Frequency Domain Methods

These methods are based on the idea of converting the signal into a time-variant spectrum and using a procedure to identify the harmonic structure, and thus the fundamental frequency, of the spectrum. One of the oldest methods is the **Cepstrum** technique [Noll, 1967]. This is considered to a component of the general method of *homomorphic speech processing*. It consists of taking the magnitude of the Fourier transform of the log of the magnitude of the signal's spectrum. Under certain conditions, the peak value of the result shows up clearly as the period of the input signal. However, this is a computationally intensive method and is "fooled" by pathological signals as much as many other methods.

M.R. Schroeder [1968] has discussed two methods used at Bell Telephone Laboratories based on the spectrum, the **period histogram** method and the **product spectrum** method. With the histogram approach, peak frequencies of the spectrum are first identified, and then all possible integral divisors of the peak frequencies are identified. These are candidate fundamentals. A histogram of the potential candidates are made, and the candidate with the most votes is chosen. This method is suseptable to spurious octave errors because a suboctave of a candidate is always a candidate. The product spectrum approach is similar in that it sums the log magnitudes of several compressions of the spectrum (i.e., $S(j\omega)$ becomes $S(jr\omega)$, for r = 1, 2, 3,...). Like the period histogram, the correct fundamental is emphasized by this process, and usually can be correctly identified.

Note that frequency domain techniques are especially appropriate if non-pitch-synchronous analysis is performed in this first place. This is the case with the McAulay-Quatieri method, but not with the heterodyne-filter/phase-vocoder approach, which for analysis purposes, requires that the analysis base frequency be close the signal's actual fundamental.

4.4.3 Some General Considerations for Pitch Detection

There are some problems and solutions shared by almost any pitch detection method. One problem is in being able to handle a wide range of pitches. Another problem is in dealing with virtuosic performance -- the faster the notes are performed, the more difficult it is to detect their pitches. Even the human ear can have difficulties. A technique which may enhance performance is *tracking*, a method based on the assumption that the value of pitch at one instant is not likely to be drastically different than its value at the next instant. Another technique is have two or more levels of pitch judging based on rank-ordered candidates.

Pitch detectors work better over narrower frequency ranges. The wider the range, the more difficult it is to achieve a high degree of success. For one thing, most pitch detectors are prone to making octave errors. Also, the level of difficulty and the best strategies for detection generally vary with the pitch register. Since there are more samples within a period for low fundamental frequencies, one might assume that correct detection of low fundamentals is easier than detection of high ones. However, the reverse is probably true. One problem with low-pitched sounds is that they are more likely to have weak fundamental components. Another problem is that there is likely to be fewer periods within each note. Also, the disparity between the durations of attack transients and the lengths of fundamental periods is typically less for lower pitch sounds than for higher ones. On the other hand, higher frequency tones have fewer harmonics and many more cycles in a typical note, factors which tend to enhance the quality of pitch detection.

Musical tones tend to have a limited life. Meanwhile, a typical pitch detector might give initial pitch estimates at a frame rate of 100/second. Indeed, many pitch detector algorithms will produce several rank-ordered pitch candidates during any frame interval. An immediate leap in pitch might be genuine. However, any sudden change in pitch which lasts only one or two frames would be highly suspect and would increase the probablity that one of the lower-ranked candidates has the real pitch. Therefore, one might build a constraint into the overall algorithm which would discredit candidates exhibiting wild, temporary pitch jumps lasting less than a certain duration. This would hopefully improve performance. Such a decision process, however, may force a tradeoff between the reliability of pitch detection for longer notes and the ability to handle rapid passages.

Judicious use of microphones is another way of enhancing the performance of a pitch detector. Close-miked sounds or sounds obtained via microphones internal to an instrument will generally have much lower harmonic content, noise, and reverberation than sounds obtained via distant air microphones, thus creating extremely favorable conditions for accurate pitch detection.

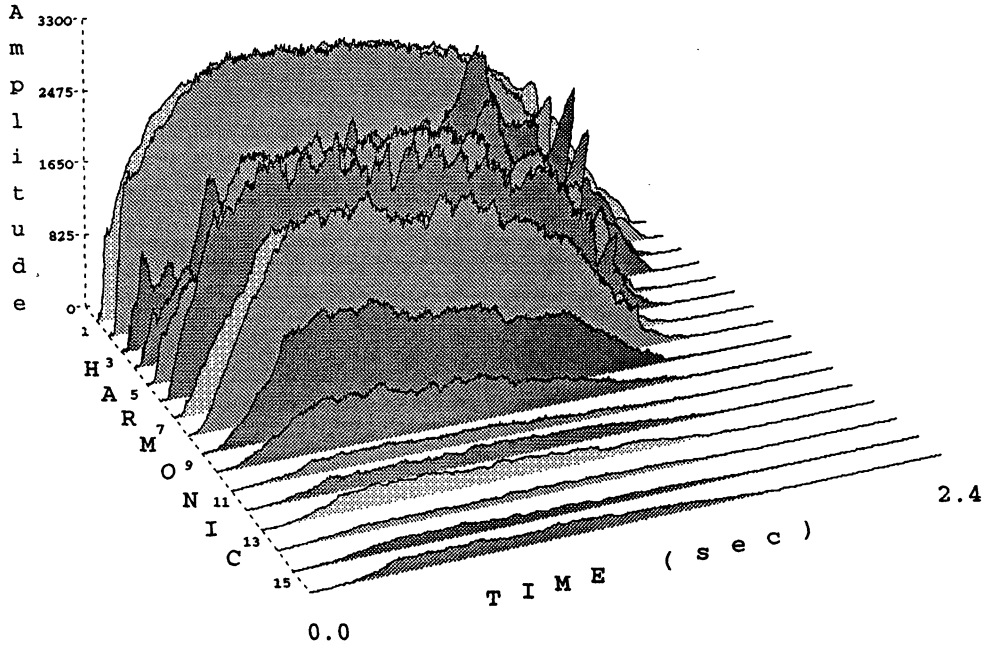
4.5 Time-Variant Spectrum Analysis Results

Once a time-variant analyzer as described has been implemented and made available to the researcher, sounds can be analyzed and resynthesized and results can be reported. Most analysis/synthesis systems have been implemented on general purpose computers, where analog-digital and digital-analog converters are used to transform data between the analog and digital domains. Graphics tools, which are available on most computers, are very useful for this type of work. Probably, the most important method of data presentation consists of plots of harmonic amplitudes (c_k) vs. time in the form of 3D perspective plots, with amplitude, frequency, and time as separate axes. Another parameter of interest is the fundamental frequency vs. time plot.

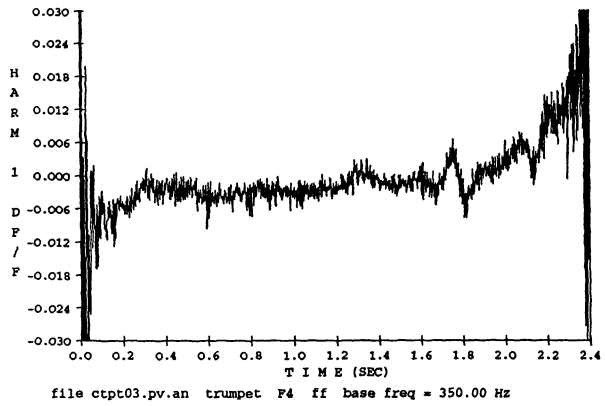
A 3D presentation of a HF/PV analysis of a trumpet tone (played *ff* at F4(350 Hz)) is shown in Figure 4.6a. Note the independence of the harmonic envelopes. However, there is a very strong tendency for upper partials to grow more slowly during the attack phase and to decay more quickly during the decay. As a result, the sound is considerably brighter (has more upper harmonics) during the loudest portion of the sound. The corresponding fundamental frequency deviation vs. time graph is shown in Figure 4.6b. The same graph is shown in Figure 4.6c with the data smoothed by a 5 Hz low pass filter. Data is presented in terms of the ratio $\Delta f/f$ for the particular harmonic graphed. Graphs for different harmonics are very similar. For this sound, we observe that there is a definite tendency for the pitch to start out below its target value and move upward at the end of the note. Frequency deviation data right at the attack and the end of the decay may be spurious because the relatively intense noise in the sound at these levels makes frequency determination less predictable.

Figure 4.7 shows the HF/PV analysis of an A2 (110Hz) guitar tone. Much of the beauty of this tone is in the attack, due to a considerable spread of energy in the upper partials for a very short amount of time. Although only 32 partials are shown, more than 120 partials are needed for full fidelity resynthesis of the sound's attack. Note again, that only the lower partials last very long.

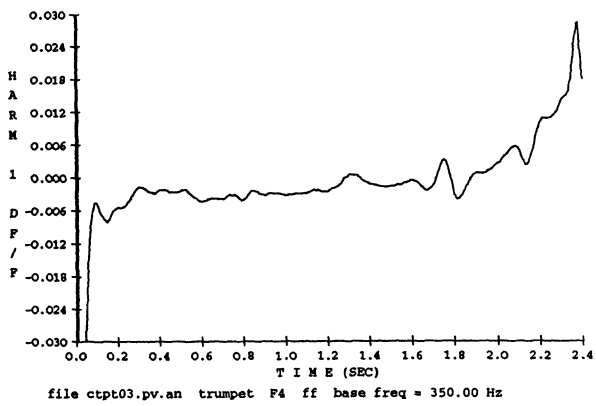
The harmonic-reduced MQ analysis of a tenor voice sound (G3, f) is shown in Figure 4.8a. MQ was needed because of the extreme variation of the fundamental frequency. The remarkable thing here is that certain harmonic amplitudes experience a considerable amount of cyclic variation which is strongly correlated with the vocal vibrato. Other harmonics, e.g., numbers 1 and 2, experience very little change. The quasi-sinusoidal vibrato waveform (i.e., the frequency deviation) is shown in Figure 4.8b. This pattern is very consistent from one partial to the next. It turns out that the apparently complex behavior of the harmonic amplitudes is strictly a result of the interaction of the tone's vibrato pattern with the voice's vocal filter response. Figure 4.9, which plots instantaneous amplitude vs. frequency for each of 48 partials, demonstrates this fact and confirms that the voice is a multiresonant subtractive synthesizer.



(a)



(b)



(c)

Figure 4.6 Heterdyne-Filter/Phase Vocoder Analysis of a Trumpet Tone (F4, 350 Hz, ff).
a) Amplitude and harmonic vs. time. b) Normalized frequency deviation ($\Delta f/f$) vs. time.
c) Smoothed version of b).

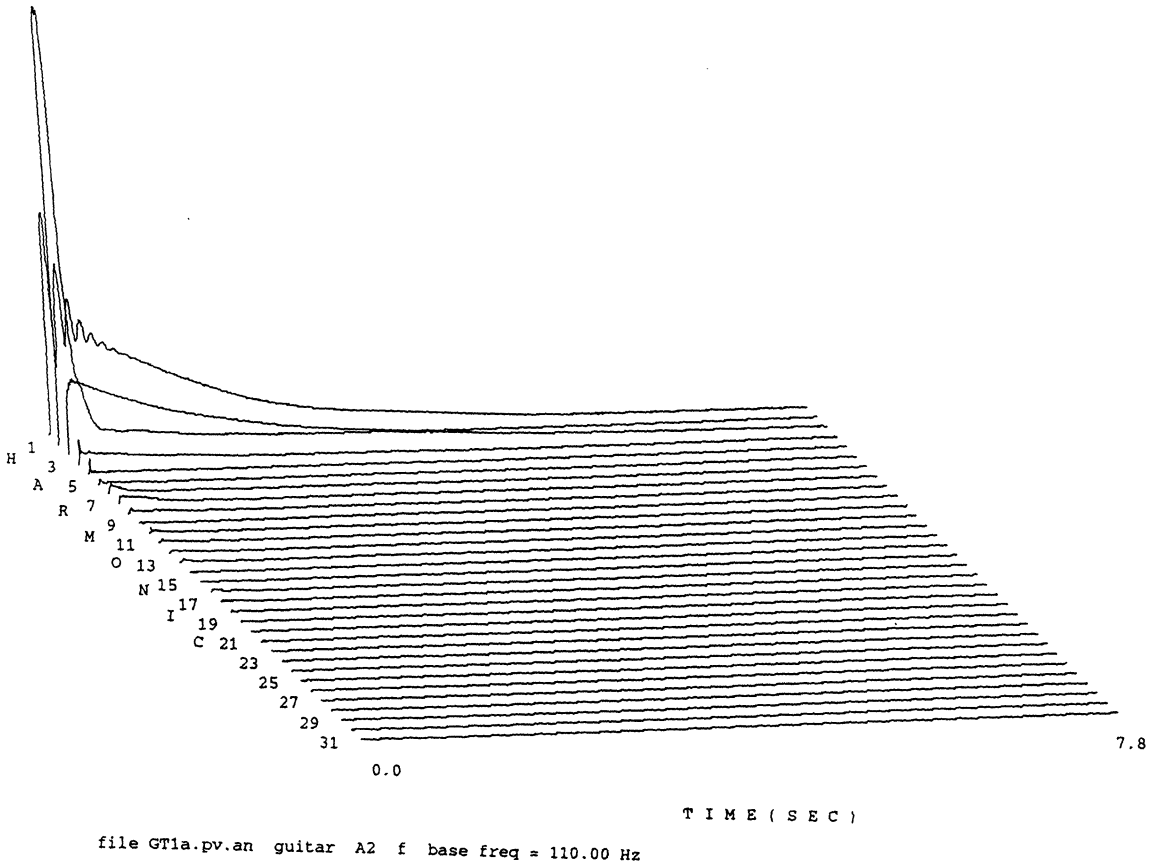
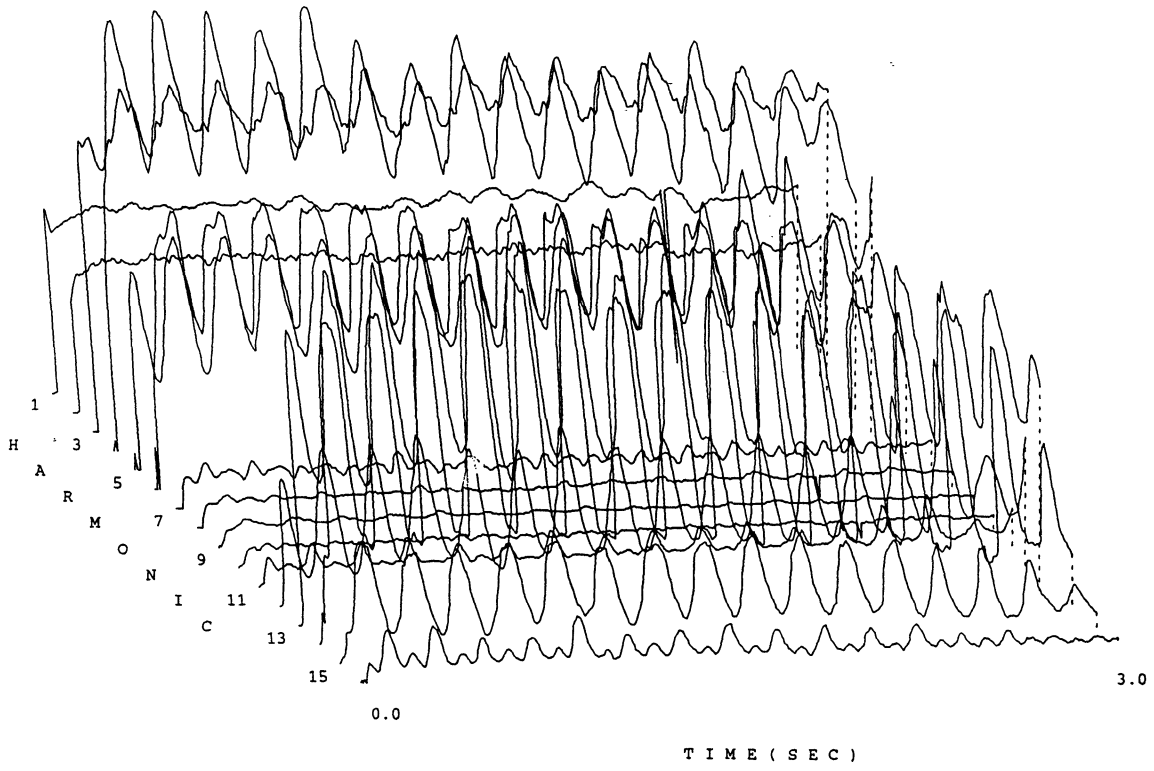
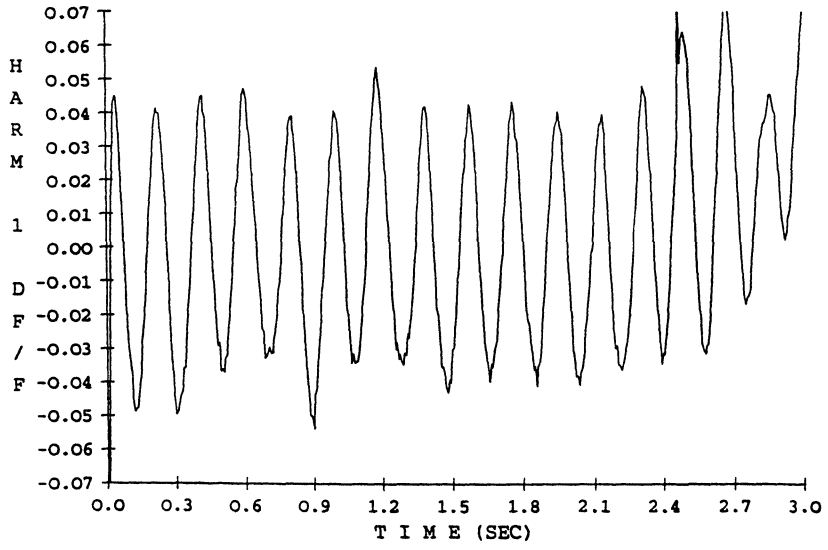


Figure 4.7 Heterodyne-Filter/Phase Vocoder Analysis of a Guitar Tone (A2, 110 Hz, 7.8 sec dur, *f*) shown as Amplitude vs. Harmonic and Time 3D Plot.



file tenor.mq.an tenor voice G3 f base freq = 192.00 Hz

(a)



file tenor.mq.an tenor voice G3 f base freq = 192.00 Hz

(b)

Figure 4.8 McAulay-Quatieri Analysis of a Tenor Vocal Tone. a) 3D Plot of Amplitude vs. Harmonic and Time. b) Frequency Deviation vs. Time.

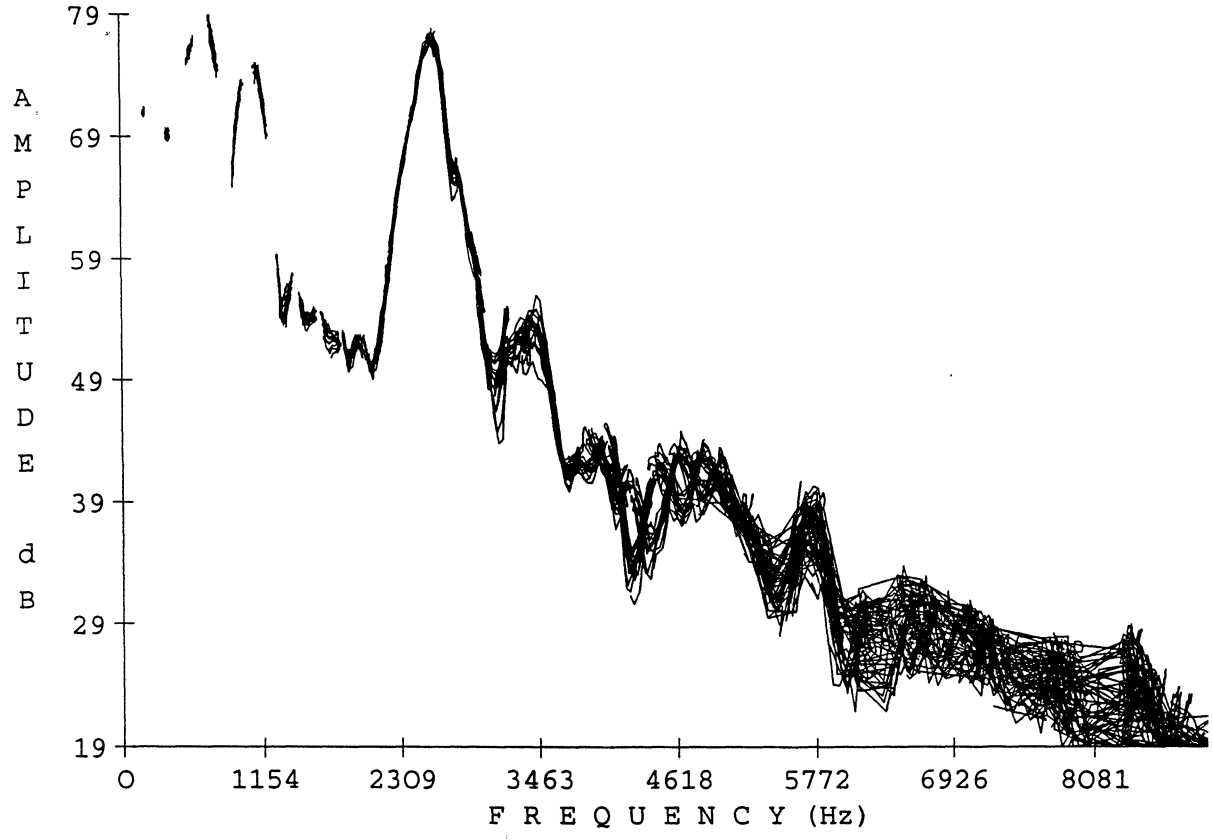


Figure 4.9 McAulay-Quatieri Analysis of Tenor Vocal Tone shown as Amplitude vs. Frequency for 48 Partials Superimposed.

References

1. J.S. Gill, "A Versatile Method for Short Term Spectrum Analysis", **Nature**, Vol. 189 (1961).

2. M.V. Mathews et al, "Pitch Synchronous Analysis of Voiced Sounds", **J. Acoust. Soc. Am.**, Vol 33, pp 179-186 (1961).

3. D.A. Luce, "Physical Correlates of Nonpercussive Musical Instrument Tones", Doctoral Dissertation, Dept. of Physics, M.I.T. (1963).

4. J.W. Beauchamp & J.P. Fornango, "Transient Analysis of Harmonic Musical Tones by Digital Computer", Audio Engr. Soc. Preprint No. 479 (1966).

5. J.L. Flanagan & R.M. Golden, "Phase Vocoder", **Bell Sys. Tech. J.**, pp 1493-1509, Nov., 1966.

6. M.D. Freedman, "Analysis of Musical Instrument Tones", **J. Acoust. Soc. Am.**, Vol. 41, pp 793-806 (1966)

7. E. Metzger, "Analysis of Musical Sounds by Fourier Transform Methods", **J. Acoust. Soc. Am.**, Vol. 42 (1967).

8. A.M. Noll, "Cepstrum Pitch Determination", **J. Acoust. Soc. Am.**, Vol. 41, pp. 293-309 (1967).

9. M.R. Schroeder, "Period histogram and product spectrum: new methods for fundamental frequency measurement", **J. Acoust. Soc. Am.**, Vol. 43, No. 4, pp. 829-834 (1968).

10. J.W. Beauchamp, "A Computer System for Time-Variant Harmonic Analysis and Synthesis of Musical Tones", **Music by Computers**, H.F. von Foerster & J.W. Beauchamp, eds, McGraw Hill, pp 19 - 62 (1969).

11. J-C Risset & M.V. Mathews, "Analysis of Musical-Instrument Tones", **Physics Today**, Vol. 22, No. 2, pp. 23-30 (1969).

12. T.A. Brubaker and H. Levin, "A Note on the Spectral Analysis of Periodic Functions", **IEEE Trans. Audio Electroacoustics**, Mar., 1972.

13. J.S. Keeler, "Piecewise-Periodic Analysis of Almost-Periodic Sounds and Musical Transients", **IEEE Trans. Audio Electroacoustics**, Vol. AU-20, pp 338-344 (1972).

14. J. A. Moorer, "The Heterodyne Filter as a Tool for Analysis of Transient Waveforms", Report No. STAN-CS-73-379, Computer Science Dept., Stanford Univ., Stanford, CA (1973).

15. J.W. Beauchamp, "Time-Variant Spectra of Violin Tones", **J. Acoust. Soc. Am.**,Vol. 56, pp 995-1004 (1974).

16. D.C. Rife, "Single-Tone Parameter Estimation from Discrete-Time Observations", **IEEE Trans. Inf. Theory**, Vol. IT-20, pp 591-598 (1974).

17. J.A. Moorer, "The Optimum Comb Method of Pitch Period Analysis of Continuous Digitzed Speech", **IEEE Trans. Acoust., Speech, and Sig. Proc.**, Vol. ASSP-22, No. 5, pp. 330-338 (1974).

18. M.J. Ross, et al, "Average Magnitude Difference Function Pitch Extractor", **IEEE Trans. Acoust., Speech, and Sig. Proc.**, Vol. ASSP-22, No. 5, pp. 330-338 (1974).

19. Bariaux, et al, "A Method for Spectral Analysis of Musical Sounds, Descriptions and Performances", **Acustica**, Vol. 32 , pp 307-313 (1975).

20. J.W. Beauchamp, "Analysis and Synthesis of Cornet Tones Using Nonlinear Interharmonic Relationships", **J. Audio Engr. Soc.**, Vol. 23, pp 778-795 (1975).

21. R.D. Weyer, "Time-Frequency Structures in the Attack Transients of Piano and Harpischord Sounds-I & II", **Acustica**, Vol. 35, pp 232-253; Vol. 36, pp 241-258 (1976/7).

22. J.M. Grey & J.A. Moorer, "Perceptual Evaluations of Synthesized Musical Instrument Tones", **J. Acoust. Soc. Am.**, Vol. 62, pp 454-462 (1977).

23. J.M. Grey, "Multidimensional Perceptual Scaling of Musical Timbres", **J. Acoust. Soc. Am.**, Vol. 61, pp 1270-1277 (1977).

24. K. Yamaguchi & S. Ando, "Analysis of Natural Musical Instrument Tones Using Digital Processing Techniques", **J. Acoust. Soc. Japan**, Vol. 33, pp 233-241 (1977).

25. J.A. Moorer, "The Use of the Phase Vocoder in Computer Music Applications",**J. Audio Engr. Soc.**, Vol. 24, pp 717-727 (1978).

26. J.L. Flanagan, "Parametric Coding of Speech Spectra", **J. Acoust. Soc. Am.**, Vol. 77, pp 412-430 (1980).

27. M.R. Portnoff, "Time-Frequency Representation of Digital Signals and Systems and Systems Based on Short-Time Fourier Analysis", **IEEE Trans. Acoust.,Speech, and Signal Processing**, Vol. ASSP-28, pp 55-59 (1980).

28. Gerard Charbonneau, "Timbre and the Perceptual Effects of Three Types of Data Reduction", **Computer Music J.**, Vol. 5, No. 2, pp 10-19 (1981).

29. J-C Risset & D.L. Wessel, "Exploration of Timbre by Analysis and Synthesis",**The Psychology of Music**, Diana Deutch, ed.; Academic Press, pp. 25-58 (1982).

30. J.O. Smith and P. Gossett, "A Flexible Sampling-Rate Conversion Method", **Proc. IEEE Conf. Acoust., Speech, and Signal Processing**, Vol. 2, pp.19.4.1-19.4.2, San Diego, March, 1984.

31. Mark Dolson, "The Phase Vocoder: A Tutorial", **Computer Music J.**, Vol. 10, No. 4, pp 14-27 (1986).

32. R.J. McAulay and T.F. Quatieri, "Magnitude-Only Reconstruction using a Sinusoidal Speech Model", **Proc. Int. Conf. Acoust., Speech, Sig. Proc.**, p. 27.6.1, San Diego, CA (1984).

33. R.J. McAulay and T.F. Quatieri, "Speech Analysis/Synthesis Based on a Sinusoidal Representation", **IEEE Trans. Acoust., Speech, Sig. Proc.**, Vol. ASSP-34, No. 4, pp. 744-754 (1986).

34. T.F. Quatieri and R.J. McAulay, "Speech Transformations Based on a Sinusoidal Representation", **IEEE Trans. Acoust., Speech, Sig. Proc.**, Vol. ASSP-34, No. 6, pp.1449-1464 (1986).

35. J.O. Smith and X. Serra, "PARSHL: An Analysis/Synthesis Program for Non-Harmonic Sounds Based on a Sinusoidal Representation", **CCRMA/Dept. of Music Rpt.**, No. STAN-M-43 (1987).

36. J. Strawn, "Analysis and Synthesis of Musical Transitions Using the Discrete Short-Time Fourier Transform", **J. Audio Engr. Soc.**, Vol. 35, pp 3-13 (1987).

37. X. Serra and J.O. Smith, "Spectral Modeling Synthesis: A Sound Analysis/Synthesis System Based on a Deterministic plus Stochastic Decomposition", **Computer Music J.**, Vol. 14, No. 4, pp. 12-24 (1990).

38. R.C. Maher, "Evaluation of a Method for Separating Digitized Duet Signals", **J. Audio Engr. Soc.**, Vol. 38, No. 12, pp. 956-979 (1990).

39. R.C. Maher and J.W. Beauchamp, "An Investigation of Vocal Vibrato for Synthesis", **Applied Acoustics**, Vol. 30, pp. 219-245 (1990).

Dynamic Spectrum Synthesis: Variable-Parameter Filters, Modulation and Nonlinear Techniques

CONTENTS

DYNAMIC SPECTRUM SYNTHESIS: VARIABLE-PARAMETER FILTERS, MODULATION AND NONLINEAR TECHNIQUES

5.0 Introduction

In the early 1960's it became clear that fixed waveforms with simple envelopes were inadequate for music synthesis. There was a general awareness of the need for varying the spectrum over time. Therefore, synthesizer designers began concentrating on methods for generating **dynamic spectra** which would evolve through several stages during the courses of sounds. Also, the assumption that partials must be purely harmonic (another consequence of periodicity) was not always correct, particularly for percussion sounds, and efficient methods for generating inharmonic spectra needed to be developed.

Two studies carried out in the early 1960's showed that *attacks* of instrument sounds are important for their recognition (e.g., Clark et al, 1963; Saldanha and Corso, 1964). Also, it became apparent for many instruments, that there is a strong connection between total amplitude and spectrum shape (Luce and Clark, 1967; Luce, 1975; Beauchamp, 1975). Therefore, any successful technique should be able to create attack features and spectrum-vs.-amplitude relationships appropriate for a variety of sounds.

Several psychoacoustic tests in the 1970's identified "brightness" (or "sharpness") as an important perceptual characteristic of tone quality. von Bismarck [1974] found that subjects listening to synthetic sounds were able to distinguish sharpness as a perceptual feature of steady sounds distinct from pitch and loudness and that sharpness was associated with spectral envelope slope (rolloff). Later tests of perceptual differences among musical instrument sounds associated the degree of brightness with the **centroid** of the spectrum [Ehresman and Wessel, 1978].

Spectral centroid can be defined many different ways. Probably the simplest definition, based on the spectral frequencies and amplitudes f_k and c_k, is:

$$f_{cg}(t) = \frac{\sum_{k=1}^{n} f_k c_k(t)}{\sum_{k=1}^{n} c_k(t)} \qquad (5.0.1a)$$

which can be thought of as the frequency in the "center" of the spectrum. For the harmonic case, this frequency translated into harmonic number, can be referred to as the *normalized spectral centroid* or "**brightness**", i.e.,

$$BR(t) = f_{cg}(t)/f_1 \qquad (5.0.1b)$$

If the c_k vary with time, it follows that f_{cg} and BR must also vary with time.

Also, the overall rms amplitude of a sound defined by

$$RMS(t) = \text{sqrt}[\sum_{k=1}^{n} c_k^2(t)\,] \qquad (5.0.2)$$

is expected to vary with time.

To demonstrate the dependence of spectral centroid on rms amplitude, we can plot BR vs. RMS for a given sound or over a group of sounds. Typically, we will obtain a curve something like the one shown below, which was plotted for a midrange trumpet tone:

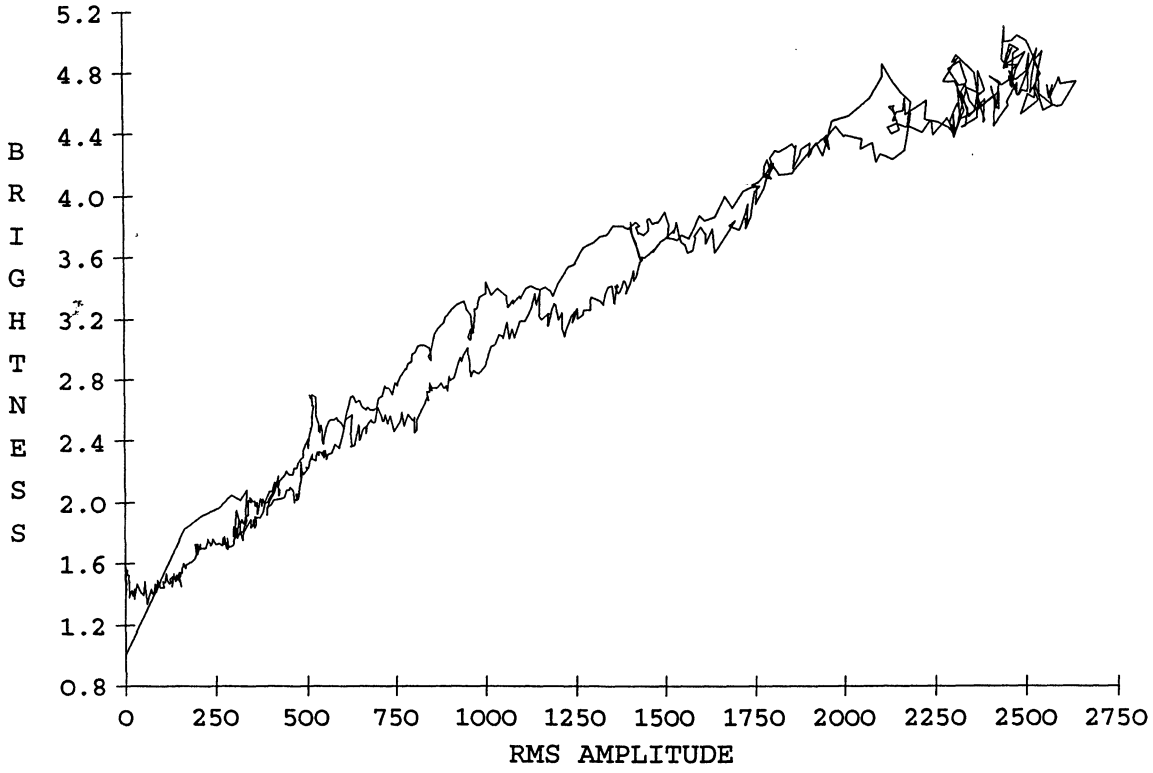


Figure 5.0.1 Normalized Spectral Centroid ("Brightness") vs. RMS Amplitude for a Midrange Trumpet Sound (F4, pp<ff>pp, .05 sec attack excluded).

"Brightness" is only a gross feature of the spectrum, but an important one, and any dynamic spectrum synthesis technique should include the ability to change "brightness" as a function of time or as a function of amplitude.

Some techniques (algorithms) for dynamic synthesis which have been successful are:

- Time-variant additive synthesis
- Variable-parameter filter synthesis
- Frequency modulation (FM) synthesis
- Nonlinear (waveshaping) synthesis
- Multiple wavetable interpolation synthesis
- Direct waveform sampling synthesis
- Transient comb filter synthesis (Karplus-Strong)
- Physical modelling synthesis

5.1 Dynamic Synthesis with Filters

5.1.1 Simple Filters with Time-Variant Parameters

Robert A. Moog introduced the voltage-controlled (VCF) filter module for his Series 900 synthesizer in 1965, and its use in studio applications soon became very popular. Beginning with "Switched-On Bach" (W. Carlos, 1968), many commercial recordings were produced which made prominant use of the VCF. This was made possible by use of an envelope generator (EG) or a low-frequency oscillator (LFO) to vary the cutoff frequency of the filter in real time. A typical patch is shown below:

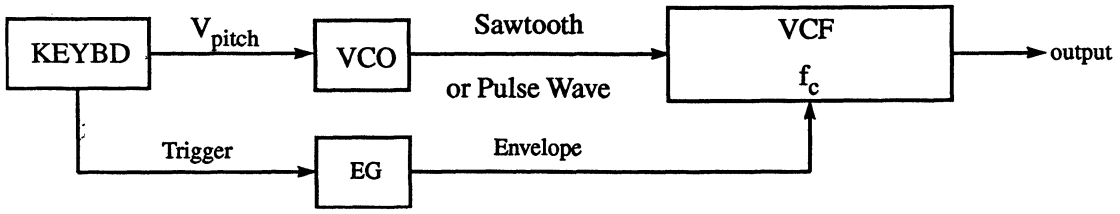


Figure 5.1.1.1 Synthesizer Block Diagram using a Voltage-Controlled Filter for Dynamic Spectrum Control

Generally, a low pass filter, with provision for resonance at its cutoff frequency, is used. If the fundamental frequency of the input is held fixed and the cutoff frequency f_c is gradually raised, the strengths of the upper harmonics gradually increase relative to the lower ones, increasing the sound's brightness. This effect can be accomplished using many different analog filter response functions.

Here are some possibilities:

$$H(f) = \frac{1}{\sqrt{1 + (f/f_c)^{2n}}}$$ nth order Butterworth

$$H(f) = \frac{1}{\sqrt{(1 - (f/f_c)^2)^2 + (1/Q)^2 (f/f_c)^2}}$$ variable Q second order low pass

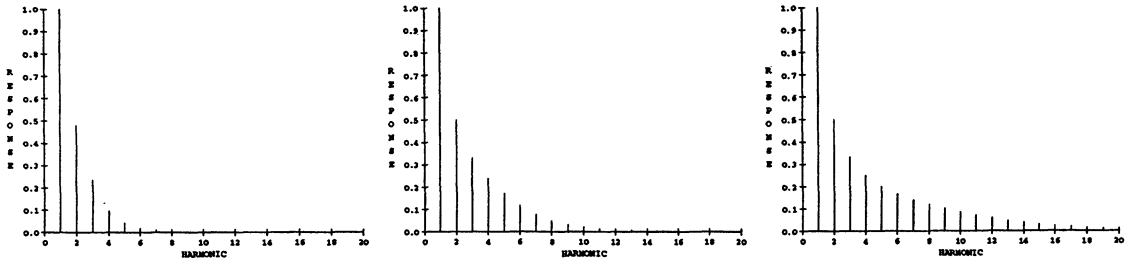
$$H(f) = \frac{1}{\left| R + (1 + jf/f_c)^4 \right|}$$ Moog 4th order low pass

The last response characterizes the Moog filter, introduced in the late 1960's. It was realized as a series of buffered RC stages, each with cutoff frequency f_c, with a negative feedback of gain R from the output to the input. It acts as a low pass filter if R=0 (no feedback), albeit without the sharp cutoff characterisitc afforded by the 4th order Butterworth. However, as R is increased, two things happen: The gain at zero frequency drops, and for R > .36 the gain at a frequency below f_c increases, causing a resonance effect. It can be shown that infinite peaking (oscillation) at $f = f_c$ results when R=4.

Given its response function, a filter's effect on an input spectrum is easy to calculate. If the input harmonic spectrum is given by $\{d_k\}$, and the fundamental frequency is f_1, the output spectrum amplitudes are given by

$$c_k = H(kf_1)\, d_k \qquad (5.1.1)$$

All of the filter responses above are functions of the ratio f/f_c. Therefore, the output spectrum is a function of f_c/f_1, and f_c is the cutoff frequency of the spectrum. Further, we can compute BR and RMS as functions of f_c/f_1 and thus as functions of each other. For variable-parameter (e.g., voltage-controlled) synthesis, f_c varies with time and, therefore, so does the output spectrum, as shown below for a 6th order Butterworth filter with sawtooth input:



$f_c = 3\, f_1$ $f_c = 6\, f_1$ $f_c = 12\, f_1$

Figure 5.1.1.2 Output spectrum from a 6th order Butterworth low pass filter driven by a sawtooth wave with variable ratio of cutoff to fundamental frequency.

5.1.2 **Complex Time-Varying Filter Synthesis: Linear Predictive Coding**

Linear predictive coding (LPC) was introduced as a speech analysis/ synthesis technique in the early 1970's [Atal and Hanauer, 1971] based on digital filters. Using z^{-1} to represent a delay of one sample and z^{-k} to represent a delay of k samples, a transfer function H(z) of the following form is used for speech synthesis:

$$H(z) = \frac{G}{1 - \sum_{k=1}^{p} a_k z^{-k}} = \frac{G}{(1 - r_1 z^{-1}) \ldots (1 - r_p z^{-1})} \qquad (5.1.2.1)$$

where the a_k coefficients and G vary over time. Note that H(z) represents an all-pole filter, and the pole positions in the unit circle are given by the complex radii $r_1, \ldots, r_p$.

A block diagram for the LPC method of synthesis is shown below:

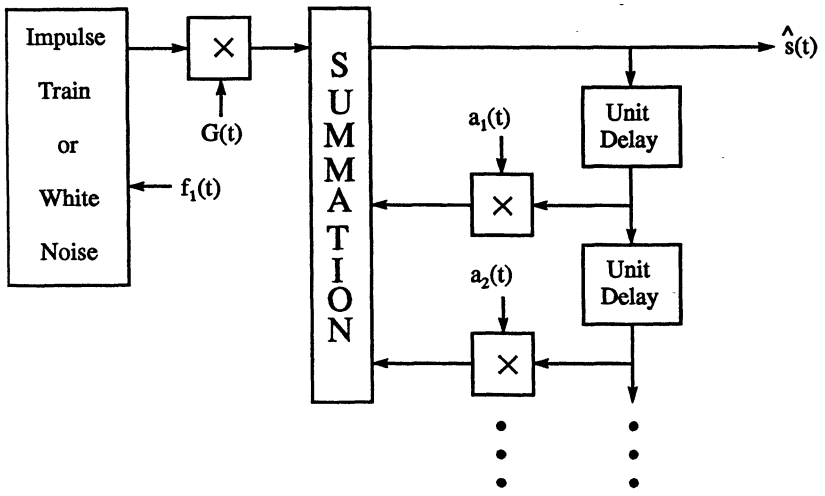


Figure 5.1.2.1 Linear Prediction All-Pole Delay Filter Synthesis

Since the LPC method is designed for sampled signals, time is quantized to an integer multiple of the sample period, i.e., $t_n = n/fs$. However, G and each a_k are held fixed during each of many successive periods called "frames". The number of unit sample delays, p, (each corresponding to one of the a_k coefficients) determines the degree of accuracy of the synthesis. The a_k's are calculated by a method of least squares minimization based on $N >> p$ consecutive samples of an acoustical signal, e.g., speech. Generally p=14 gives good results for speech sampled at f_s = 10 KHz. The length of a frame, N/f_s, can vary anywhere from $2p/f_s$ to 40 milliseconds, depending on the method of LPC analysis used. LPC synthesis is quite efficient for speech and has been used in hardware devices like Texas Instrument's "Speak and Spell" toy. During a single frame the following recursion formula is used to generate a series of samples:

$$\hat{s}_n = G\,\delta(n) + \sum_{k=1}^{p} a_k \hat{s}_{n-k}, \quad n = 0, 1, \ldots, N-1 \qquad (5.1.2.2)$$

where n = 0 refers to the first sample of the frame and n = N-1 is the last sample; $\delta(n)$ is the unit impulse, which is 1 for n=0, otherwise it is zero. If the signal is just starting up, negative indices give zero sample values, but generally they reference sample values from the previous frame. At this point we note that samples of an original signal can be exactly regenerated by Equation 5.1.2.2 provided $G\,\delta(n)$ is replaced by a signal $\mathbf{e_n}$, called the "residual signal". Thus,

$$s_n = e_n + \sum_{k=1}^{p} a_k s_{n-k} \qquad (5.1.2.3)$$

where the e_n are given by

$$e_n = s_n - \sum_{k=1}^{p} a_k s_{n-k} \qquad (5.1.2.4)$$

It might seem like we are going around in circles, but this actually leads to a method of determining appropriate values for the a_k's. We can think of the nth sample of the signal, s_n, as being *predicted* by a linear combination of previous samples. To do this, we attempt to find the best values of the a_k's in order

to minimize the average value of e_n over a series of samples. The minimization is performed by first adding up the squared values of e_n over the frame duration of N samples:

$$E = \sum_{n=0}^{N-1} e_n^2 = \sum_{n=0}^{N-1} (s_n - \sum_{k=1}^{p} a_k s_{n-k})^2 \qquad (5.1.2.5)$$

We then take derivatives with respect to each a_k ($1 \le k \le p$) and set the p equations to zero. Since this yields p linear equations in p unknowns, we can solve for the a_k values which minimize E in the least squared sense. So far in our discussion, e_n has just been a hypothetical signal.; however, once we have determined the a_k's, we can compute the actual residual signal by repeating the recursion formula of Equation 5.1.2.4. If this signal were used as input to the LPC filter, the original signal would be reproduced exactly, as Equation 5.1.2.3 clearly states. However, this would be no reduction of data. The trick is to replace the residue e_n with either an impulse train signal (for voiced sounds) or white noise (for unvoiced sounds). Happily for speech signals, the residue resembles a pulse train when speech is voiced, with the pulses spaced according to the pitch period of the voice, and like white noise during the unvoiced sounds. Both of these signal forms tend to be spectrally flat, a natural outcome of the least-squares minimization process.

Charles Dodge has used the LPC technique, implemented on general purpose computers, to produce music compositions based on speech input [Dodge, 1985]. With his method, spoken poetry is digitized at 15 KHz and analyzed to produce 24 parameters for each frame at a rate of 120 frames per second of the original speech. The first 3 parameters are *low-frequency amplitude*, *high-frequency amplitude*, and *errn*, which are used to determine whether the speech sample is voiced (pulse train input) or unvoiced (white noise input). The other 21 parameters consist of the frame duration, the fundamental frequency (relevant only if voiced), and the 19 coefficients of an all-pole digital filter. With Dodge's program it is possible to alter duration and pitch independently, and to repeat and overlay (mix) segments. Other composers who have used this technique are Joseph Olive, Paul Lansky, and J.A. Moore.

5.2 Modulation Synthesis

Modulation synthesis refers to a class of methods which are capable of producing complex spectra whose properties change in response to variations of one or more parameters. In general, a modulating signal (sometimes called the "program") is used to vary some parameter of a signal (called the "carrier"). Generally, if the modulating signal is a harmonic signal and the carrier is a sine wave, the synthetic spectrum will consist of components spaced according to fundamental frequency of the modulating signal, centered at the frequency of the carrier. The resulting spectrum may be either harmonic or inharmonic and generally will also depend on the amplitude and spectral envelope of the modulating signal.

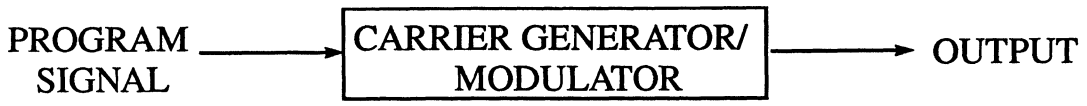


Figure 5.2.1 Generalized Modulation Synthesis Block Diagrm

The principal modulation synthesis techniques which have been used in electronic and computer music are amplitude modulation (AM) and frequency modulation (FM).

5.2.1 Amplitude Modulation

Amplitude modulation may be thought of as the multiplication of two or more signals. For two signals the general input/output equation is given by

$$s_{out}(t) = s_1(t) \bullet s_2(t) \qquad (5.2.1.1)$$

In ordinary AM radio transmission $s_1(t)$ is an audio program with 7 KHz bandwidth and $s_2(t)$ is a sine wave carrier signal, i.e., $s_2(t) = \cos(2\pi f_c t)$, where fc is between 550 and 1800 KHz. $s_1(t)$ is assumed to be always positive. If it varies between s_{1max} and s_{1min}, the *percent modulation* is given by

$$\%\text{mod} = (s_{1max} - s_{1min}) \times 100 / s_{1max} \qquad (5.2.1.2)$$

For music applications, *balanced amplitude modulation*, where the average (DC) value of $s_1(t)$ is zero, is much more useful. Unlike the unbalanced case, the spectrum of the balanced modulator (sometimes called "ring modulator") output is theoretically devoid of carrier component, and consists only of two *side band* spectra, an *upper side band* and its mirror image, the *lower side band.* Consider the case where $s_2(t) = \cos(2\pi 1000t)$ and $s_1(t) = \cos(2\pi 114t)$. The output signal is given by

$$s_{out}(t) = .5\cos(2\pi 886t) + .5\cos(2\pi 1114t) \qquad (5.2.1.3)$$

where 886 Hz and 1114 Hz are the lower and upper sideband components, respectively, as shown below:

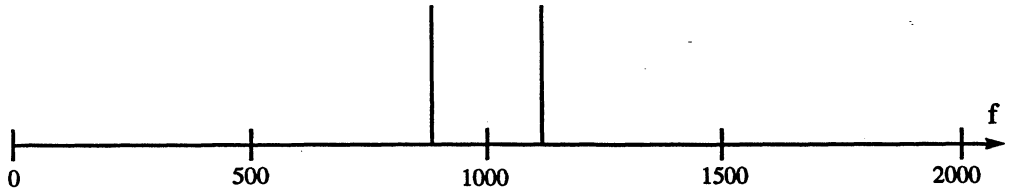


Figure 5.2.1.1 Effect of balanced amplitude modulation of a 1000 Hz sine wave by a 114 Hz sine wave shown in terms of frequency spectra.

Now, consider the case where $s_2(t)$ is still a 1000 Hz sine wave but $s_1(t)$ consists of the first 5 components of a sawtooth wave ($A_k = 1/k$). The output spectrum now looks like:

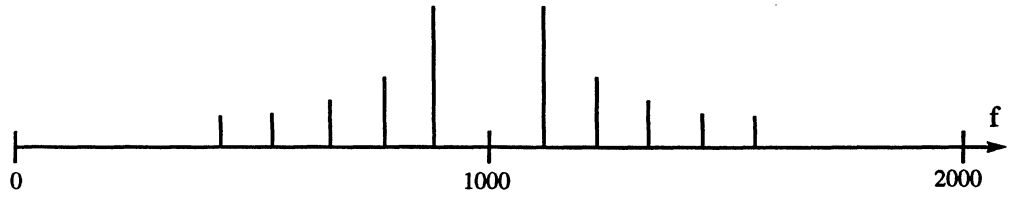


Figure 5.2.1.2 Effect of balanced amplitude modulation of a 1000 Hz sine wave by a complex wave shown in terms of frequency spectra.

It can be seen that the upper and lower side bands are mirror images of each other, unless one or more of the components of s_1 has a frequency greater than the carrier. If it does, foldover occurs about the zero frequency axis, altering the symmetry.

Single side band amplitude modulation spectra can be generated if both sine and cosine components of the carrier and program signals are available. In terms of steady-state spectra we can define *quadrature* versions of $s_1(t)$ and $s_2(t)$ as follows:

$$s_1(t) = \Sigma\, a_k \cos(2\pi k f_1 t),\quad s_1'(t) = \Sigma\, a_k \sin(2\pi k f_1 t)$$

$$s_2(t) = \cos(2\pi f_c t),\quad s_2'(t) = \sin(2\pi f_c t). \qquad (5.2.1.4)$$

We then form the two products $s_1(t).s_2(t)$ and $s_1'(t).s_2'(t)$ and combine the results. Addition cancels the sum frequencies, leaving the lower side band only, while subtraction accomplishes the opposite, leaving a separated upper side band. In practice, the quadrature program signals $s_1(t)$ and $s_1'(t)$ are formed by constant 45° phase shift circuits operating on a common signal:

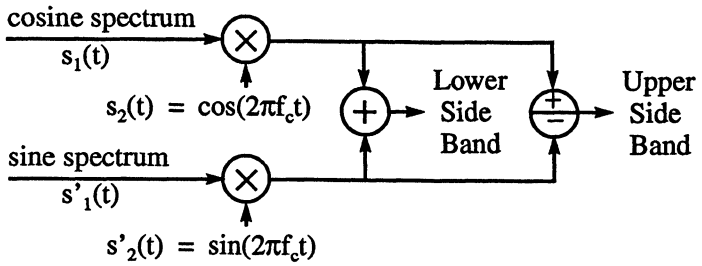


Figure 5.2.1.3 Single side band modulation synthesis

Another way to create a single side band single is to use a sharp cutoff high pass or low pass filter to select either the lower or upper side band. In the digital domain, this can be simulated using discrete Fourier transforms.

With either double or single sideband AM, interesting effects result if the program signal is a recording of an musical instrument or voice sound. The spectrum of the original signal is transformed into a new set of frequencies not related as in the original sound, thus producing a dramatic change in tone quality. On the other hand,, the attack/decay envelope of the instrumental sound is retained (amplitude modulation is linear with respect to changing amplitude of the carrier or program) and is superimposed, in effect, on the new spectrum. The result is that the modulated sound retains much of its original character even though its frequencies are shifted. When speech is used for the program signal, , amplitude-modulaion can be reminiscent of Helium speech, i.e., speech uttered in a Helium-rich environment.

5.2.2 **Frequency Modulation Synthesis**

Frequency modulation (FM) is a technique which provides "dynamic spectrum control", a control of the center-of-gravity of the acoustic spectrum as a function of a single control parameter. Compared to the variable-parameter filter technique, FM produces a similar effect, but the detailed behavior of the control is different for the two cases. As a practical matter, variable-parameter filters have been more economical for analog synthesizers, whereas frequency modulation has been more cost effective for digital implementations.

As mentioned earlier, the inspiration for dynamic spectrum control derives from the behavior of acoustical musical instrument tones, which we, as listeners of music, have become conditioned to: Practically all acoustical instrument tones increase in brightness as their intensities increase. Brightness is a subjective attribute which is related to the relative strength of the high frequency portion of the spectrum compared to the lower frequency portion, and psychoacoustic tests lead us to believe that a strong correlation exists between brightness and the center-of-gravity of the spectrum.

5.2.2.1 **Basic Theory of FM**

The simplest frequency modulation formulation, for which the analysis is well known, is sinusoidal modulation of a sine wave carrier. Let the instantaneous frequency of the sine wave be given by

$$f(t) = f_c + \Delta f \cos(2\pi f_m t), \qquad (5.2.2.1)$$

where Δf is the frequency deviation and f_m is the modulation frequency. The instantaneous phase can be obtained by integrating this expression to produce

$$\theta(t) = 2\pi f_c t + (\Delta f/f_m) \sin(2\pi f_m t). \qquad (5.2.2.2)$$

This equation shows that the normal sine wave phase term, $2\pi f_c t$, is modulated by addition of a sinusoidally-varying term with frequency f_m and phase amplitude given by

$$\alpha = \Delta f/f_m, \qquad (5.2.2.3)$$

which we call the **frequency modulation index.** Actually, we normally only plot phase on a 0 to 2π

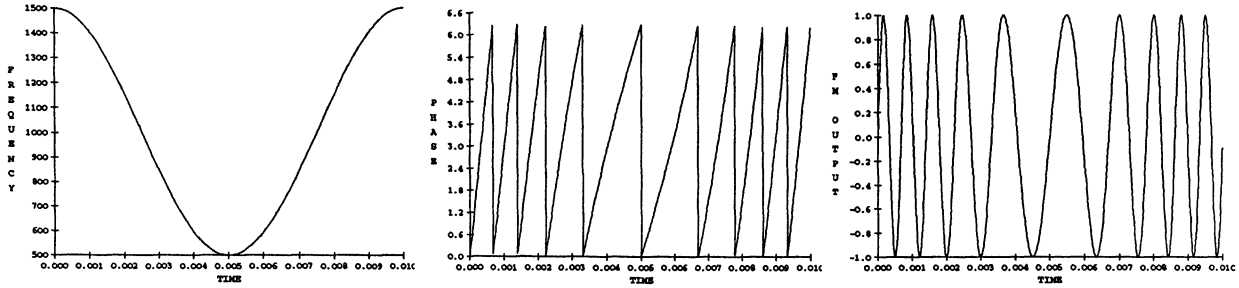
range (which in a digital implementation would correspond to a range of indices used for table lookup), so Equation 5.2.2.2 can be rewritten as

$$\theta(t) = \text{mod}\{ 2\pi f_c t + \alpha \sin(2\pi f_m t), 2\pi\}. \qquad (5.2.2.4)$$

The frequency-modulated signal is formed when this phase function is applied to the sine function:

$$s(t) = \sin(\theta(t)) = \sin(2\pi f_c t + \alpha \sin(2\pi f_m t)) \qquad (5.2.2.5)$$

Typical plots of instantaneous frequency, phase, and output waveform are shown in Figure 5.2.2.1. Obviously, the introduction of frequency (or phase) modulation profoundly affects the output, in this case by modulating the period of the carrier sine wave.



(a) (b) (c)

Figure 5.2.2.1 Frequency modulation when $f_c = 1000$, $f_m = 100$, and FM index = 5: a) frequency vs. time, b) phase vs. time, c) output waveform vs. time.

More importantly, we would like to know how the output signal's spectrum is affected. Fortunately, it can be shown that the form $\sin(x + \alpha \sin(y))$ of Equation 5.2.2.5 can be expanded in terms of Bessel functions of the first kind ($J_k(\alpha)$) as coefficients of $\sin(x \pm ky)$:

$$\sin(2\pi f_c t + \alpha \sin(2\pi f_m t)) = \sum_{k=-\infty}^{\infty} J_k(\alpha) \sin(2\pi(f_c + kf_m)t)$$

$$= J_0(\alpha) \sin(2\pi f_c t) + \sum_{k=1}^{\infty} J_k(\alpha) [\sin(2\pi(f_c + kf_m)t) + (-1)^k \sin(2\pi(f_c - kf_m)t)] \qquad (5.2.2.6)$$

Note that $J_{-k}(\alpha) = (-1)^k J_k(\alpha)$.

The J_k Bessel functions are probably the most commonly-available mathematical functions after the log, exponential, and trigonometric functions and are often used in the solution of differential equations based on circular geometries. Plots of some J_k functions for $k \geq 0$ are shown in Figure 5.2.2.2.

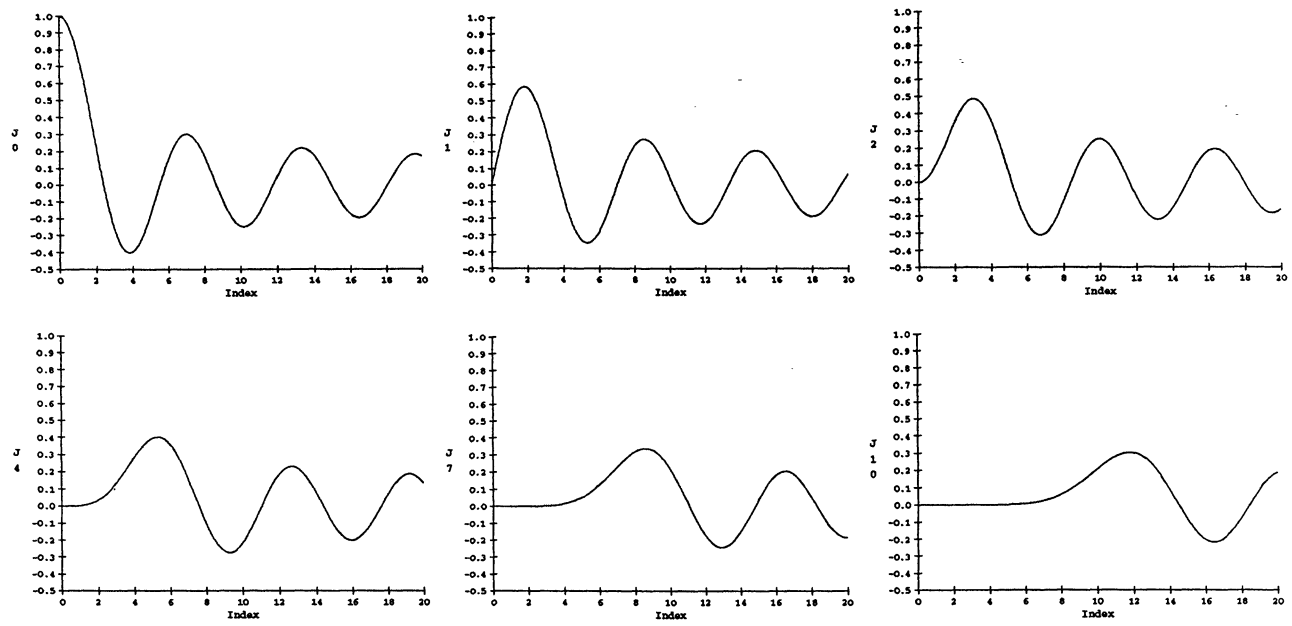


Figure 5.2.2.2 $J_k(\alpha)$ vs. α for k = 0, 1, 2, 4, 7, and 10.

Three aspects of the J_k's are obvious from the plots. One, the point at which the rise to first maximum begins is an increasing function of k. Two, all functions tend to oscillate with a period which eventually approaches 2π. Three, the general amplitude envelopes of the functions decrease with increasing values of their argument α. Their behavior can be summarized by these two asymptotic limits:

$$J_k(\alpha) \sim \frac{\alpha^k}{k!\,2^k}, \quad \alpha \to 0 \qquad (5.2.2.7\alpha)$$

$$J_k(\alpha) \sim \sqrt{\frac{2}{\pi\alpha}} \cos(\alpha - k\pi/2 - \pi/4), \quad \alpha \to \infty \qquad (5.2.2.7b)$$

For small α, J_0 (the carrier amplitude) is the largest of the J_k's and the other component amplitudes (the side band amplitudes) get smaller as k increases. As α increases, a situation is reached where the amplitudes of all side band components, up to some critical value of k, oscillate between 0 and sqrt($2/\pi\alpha$) as a function of k.

Some typical FM spectra (for α = 0.2, 1, and 5) are shown below:

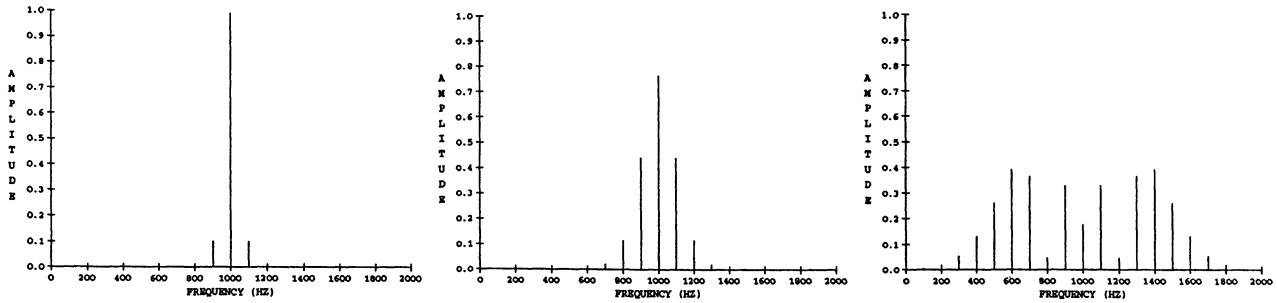


Figure 5.2.2.3 FM magnitude spectra for Index = 0.2, 1, and 5. For these examples f_m=100 and f_c=1000 Hz.

Note that as α increases, the bandwidth increases, even though the component amplitudes experience ups-and-downs due to the oscillatory nature of the Bessel functions. The total rms amplitude stays approximately the same. If we define bandwidth BW as a measure of the band containing all components with amplitudes ultimately greater than .01 (-40 dB), we can plot a curve of normalized bandwidth (BW/Δf) vs. α as shown below:

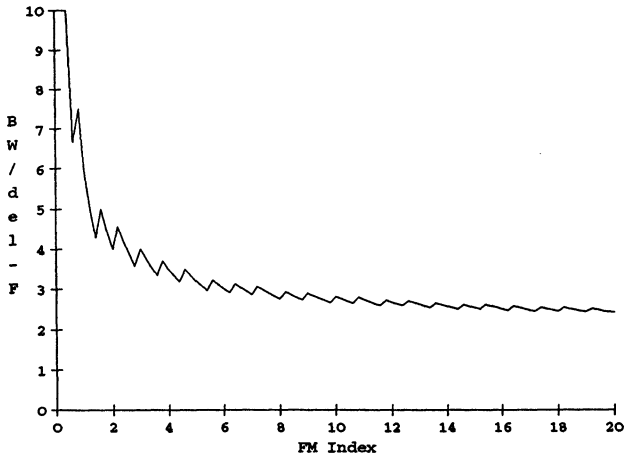


Figure 5.2.2.4 Normalized FM bandwidth vs. FM Index (α)

The vertical scale of Figure 5.2.2.4 can also be interpreted as *twice the highest sideband number over the FM Index* ($2\,k_{max}/\alpha$). While increasing α causes k_{max} to increase, it increases faster for small α than for large, in which case k_{max} assymptotically approaches α. Thus, for very slow FM, f_m is small (dense component spacing), α is large, and BW $\approx 2\Delta f$, which means that the bandwidth is confined to the range of $f_c \pm \Delta f$ -- a very intutive result. When the speed of FM increases (while Δf is held fixed), the distance between the spectral components increases and α drops. While the number of components in the band diminishes, the actual bandwidth increases beyond $2\Delta f$ according to the data of Figure 5.2.2.4.

5.2.2.2 Single-Carrier, Single-Modulator FM Technique

John Chowning [Chowning, 1973] developed an FM synthesis technique for generating various classes of sounds by selection of particular f_c-to-f_m ratios and by use of envelope generators to appropriately vary the FM index and the total amplitude of the sound. The technique utilizes the fact that if the index is sufficiently large, lower side band FM components will have *negative frequencies* which are effectively reflected as 180° phase-shifted positive-frequency components.

If the two frequencies are related by a rational multiplier, a harmonic tone results. More precisely, if $f_m/f_c = N_2/N_1$, a harmonic tone with fundamental frequency $f_1 = f_c/N_1 = f_m/N_2$ is produced.

The simplest possible case occurs when $f_m = f_c = f_1$, which has been useful for simulating brass tones. Equation 5.2.2.6 reduces to

$$s(t) = \sum_{k=-\infty}^{\infty} J_k(\alpha) \sin(2\pi(k+1)f_1 t) \qquad (5.2.2.8a)$$

which we can expand as

$$s(t) = \sum_{k=2}^{\infty} J_{-k}(\alpha)\sin(2\pi(-k+1)f_1t) + \sum_{k=0}^{\infty} J_k(\alpha)\sin(2\pi(k+1)f_1t)$$
$$= \sum_{k=1}^{\infty} -J_{-(k+1)}(\alpha)\sin(2\pi kf_1t) + \sum_{k=1}^{\infty} J_{(k-1)}(\alpha)\sin(2\pi kf_1t)$$
$$= \sum_{k=1}^{\infty} \{J_{(k-1)}(\alpha) + (-1)^k J_{(k+1)}(\alpha)\}\sin(2\pi kf_1t) \qquad (5.2.2.8b)$$

In this case, negative frequency components have reflected back to coincide with the positive ones so that each harmonic amplitude is the combination of two Bessel functions:

$$c_k = \left| J_{(k-1)}(\alpha) + (-1)^k J_{(k+1)}(\alpha) \right|$$

where the magnitude is taken because, by definition, c_k is nonnegative.

A particular index envelope is needed to create a brass-like sound, and a maximum index value of 5 was suggested by Chowning in his 1973 paper. He recommended an envelope shape similar to that given in Figure 5.2.2.4a. In addition, he gave the following recipes for simulation of some instruments:

Bassoon tone: $f_c = 500$ Hz, $f_m = 100$ Hz, $\alpha_{max} = 1.5$

Clarinet tone: $f_c = 900$ Hz, $f_m = 600$ Hz, $\alpha_{max} = 4$

Bell tone: $f_c = 200$ Hz, $f_m = 280$ Hz, $\alpha_{max} = 10$

Let us examine some reasons for these choices. First, bassoon tones typically have a resonance around 400-500 Hz. Using $\alpha_{max} = 1.5$ gives a smooth resonance for several partials which occur around the resonance center. For the values given, the fundamental frequency is 100 Hz. Second, clarinet tones are characterized by an emphasis on odd harmonics of the fundamental. Choosing $f_c/f_m = 1.5$ results in odd FM frequencies: .5 f_m, 1.5 f_m, 2.5 f_m, ... Finally, for the bell tone, the choice $f_c/f_m = 200/280 = 1/1.4$ gives an approximate inharmonic series, and the frequencies which result, 80, 200, 360, 480, 640, 760, 920 ... approximate the pitches Eb2, G3, F4, Bb4, Eb5, F5, ..., which is close to Eb major chord, actually more typical of chime than of bell sound. (An idealized chime actually would produce a series like 72, 72x25/9, 72x49/9, ... $80x(2n+1)^2$, ... or 72, 200, 392, 648, 968, ... , which bears a "striking" resemblence to the first few partials of the FM series!) Figure 5.2.2.4 shows the amplitude and index envelopes and the FM spectrum corresponding to α_{max} for each of the 4 cases discussed.

The "patch" for the simple FM instrument Chowning used is illustrated in Figure 5.2.2.5. Two envelope generator/sine wave oscillator combinations are used, one for the modulator and one for the carrier. Yamaha, Inc., in their line of FM synthesizers introduced in 1983, referred to each EG/osc. combination as an "operator". Thus, this is a simple two-operator patch.

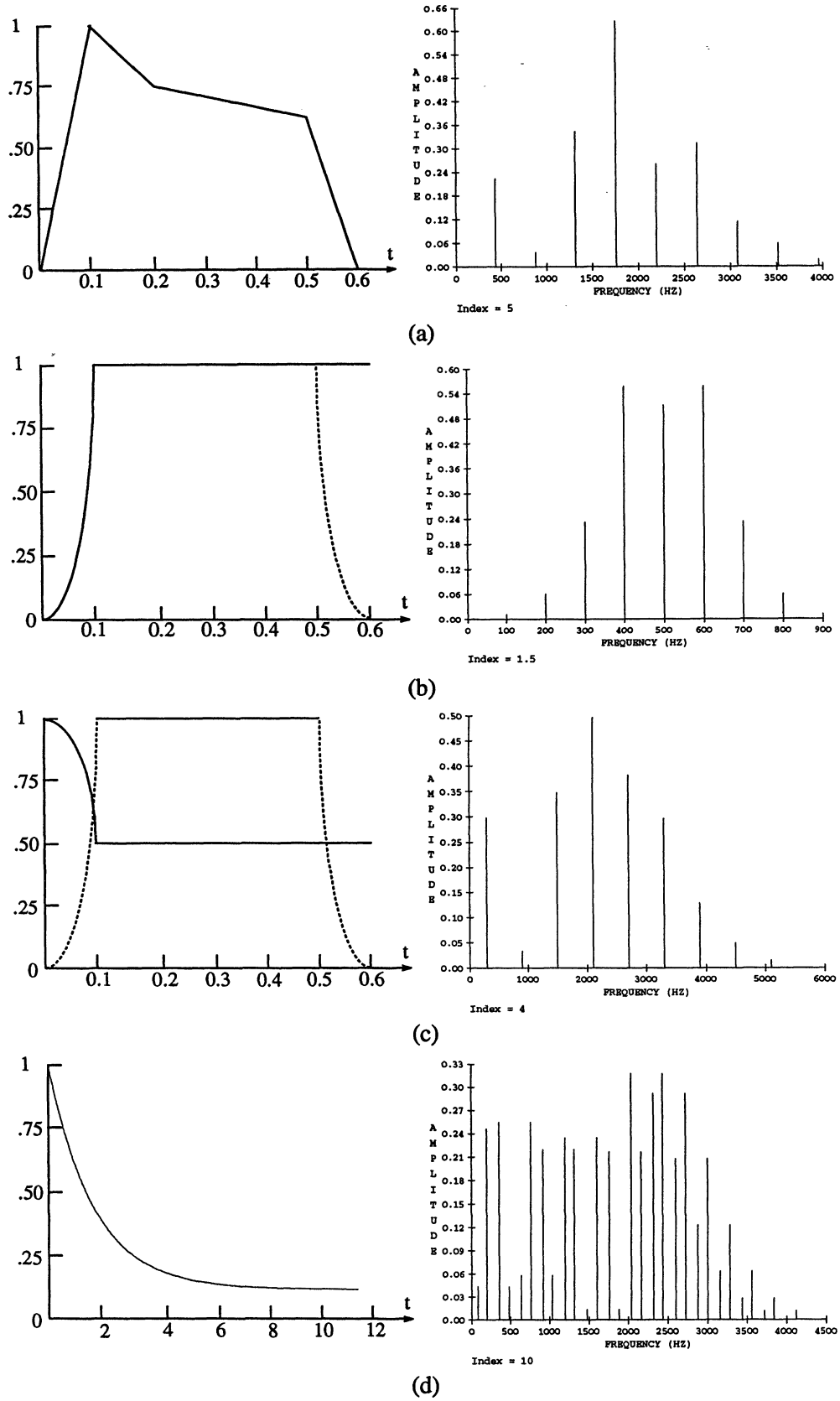


Figure 5.2.2.4 FM index (solid lines) and amplitude (dashed lines) envelope functions and maximum-index FM spectra to simulate various instruments: a) trumpet, b) bassoon, c) clarinet, d) bell (after Chowning, 1973).

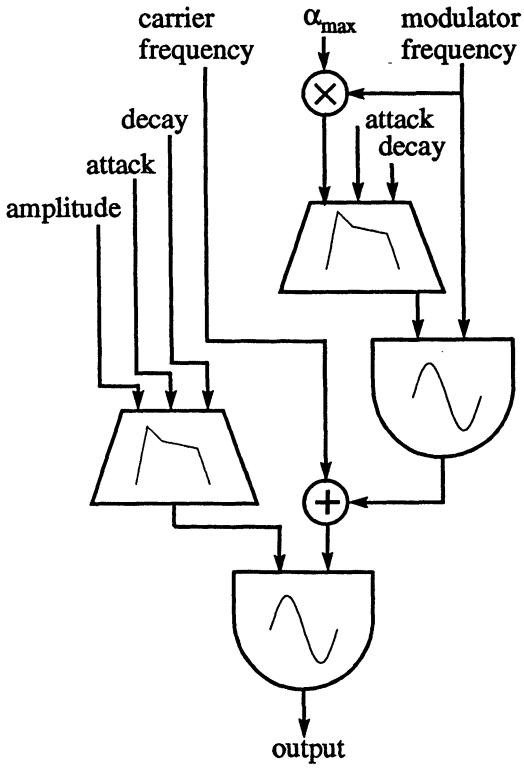


Figure 5.2.2.5 Two envelope generator/sine wave oscillator frequency modulation patch (flow diagram), which can be used to simulate a variety of instrumental sounds. Trapezoidal figures represent envelope generators and the half round figures represent oscillators.

Calculation of FM Spectra, Taking into Account Negative Frequency Components

Negative frequency components reflect about the zero frequency axis to form positive frequency components with amplitudes of the opposite sign to their original signs. This can be seen by observing that in Equation 5.2.2.6, the components represented by $(-1)^k J_k(\alpha) \sin(2\pi(f_c - kf_m)t)$ can also be written as $-(-1)^k J_k(\alpha)\sin(2\pi(kf_m - f_c)t)$. If $f_c - kf_m$ is negative, then $kf_m - f_c$ is the positive version of that frequency, and the sign tranfers to the component's amplitude. Another sign change occurs if k is odd, which is due to the $(-1)^k$ factor. Finally, $J_k(\alpha)$ itself can be positive or negative (see Figure 5.2.2.2), so there are three possibilities for sign changes. The sign of a reflected component is not important if its frequency is unique. However, often the reflected components have the same frequency as some or all of the unreflected compoents. In these cases, the reflected and unreflected components combine either constructively (same sign) or destructively (opposite signs). These reflective "collisions" will normally occur whenever a harmonic spectrum results from particular choices of carrier and modulator frequencies.

Effect of Phase and Finite Sample Rate on FM Spectra

The FM equation examined so far (Equation 5.2.2.5) is for particular carrier and modulator phases, namely $\phi_c = 0$ and $\phi_m = \pi/2$. We can write a more general form of the FM equation as

$$s(t) = \sin(\phi_c + \int_0^t [\omega_c + \Delta\omega \sin(\omega_m t + \phi_m)]dt) \qquad (5.2.2.9a)$$

$$= \sin(\omega_c t + \phi_c + \alpha(\cos(\phi_m) - \cos(\omega_m t + \phi_m))) \qquad (5.2.2.9b)$$

and it can be shown that this can be expanded as

$$\sum_{k=-\infty}^{\infty} J_{|k|}(\alpha) \sin((\omega_c + k\omega_m)t + \phi_c + \alpha\cos(\phi_m) + k\phi_m - |k|\pi/2). \qquad (5.2.2.9c)$$

Note that this reduces to Equation 5.2.2.6 when $\phi_c = 0$ and $\phi_m = \pi/2$. However, another important case is for $\phi_c = 0$ and $\phi_m = 0$, settings which are probably used more often in actual implementations of FM.

Let k_u correspond to a positive (unreflected) frequency sideband ($f_c + k_u f_m$) and k_r to a negative (reflected) frequency sideband ($f_c + k_r f_m$), and let R, an integer, be twice the carrier-to-modulator ratio. While k_u can be either positive or negative, k_r is always negative. Unreflected and reflected sidebands will combine if $f_c + k_u f_m = -(f_c + k_r f_m)$. This leads to

$$k_r = -(k_u + R) \qquad (5.2.2.10)$$

It turns out that it is useful to define a variable H, where if R is even, H = k is the harmonic number of the combined component (i.e., H = 1, 2, 3, 4, ...); however, if R is odd, 2H is an odd harmonic number, so that H = k/2 = 1/2, 3/2, 5/2, ... Also, it follows that $k_u = H - R/2$, and $k_r = -(H + R/2)$.

Then, the general formula for the amplitude of the two-component FM combinations is given by

$$c_k = \sqrt{J_{|k_u|}^2(\alpha) + J_{|k_r|}^2(\alpha) + 2J_{|k_u|}(\alpha)J_{|k_r|}(\alpha)\cos(\theta)} \qquad (5.2.2.11a)$$

where θ is given by

$$\theta = 2\phi_c + 2I\cos(\phi_m) - R\,\phi_m - (|H - R/2| + H + R/2)\,\pi/2 \pm \pi. \qquad (5.2.2.11b)$$

Test Case: Let $f_c = f_m$, $\alpha = 4$, and $\phi_c = \phi_m = 0$. Then R = 2 (R is even) and

$$\theta = 2\times 4 - 2H\pi/2 \pm \pi = 8 \pm (H+1)\pi$$

The amplitudes for the first eight harmonics are then:

0.576541, 0.425608, 0.426420, 0.468005, 0.278258, 0.135132, 0.048665, 0.015341

If we repeat this for $\phi_c = 0$, $\phi_m = \pi/2$ (90°), this spectrum becomes:

0.761278, 0.364128, 0.082999, 0.562258, 0.232041, 0.147263, 0.045059, 0.016115

This demonstrates that the carrier and modulator phases have a profound effect on the resulting FM spectrum. However, it is not clear that there is much perceptual difference as phase is changed. For more details on FM phase effect see Bate (1990) and Beauchamp (1992).

Holm (1992) gives a detailed analysis of the effect of sample rate on the FM spectrum. This effect results

from the fact that a digital implementation of Equation 5.2.2.9a reduces the integral to a discrete sum, and thus is an approximation to that equation, whose accuracy degrades as the sample rate is reduced. The solution to this problem is to instead base the digital implementation on phase modulation as given be Equation 5.2.2.9b; then there is no problem with approximation, except the usual foldover problem.

5.2.2.3 **Multiple-Carrier FM Synthesis**

Simple two-operator FM can only be made to crudely imitate live acoustic sounds. However, if more complexity is built into the model, closer matches to original sounds are possible. Early attempts were made by Morrill (1977), Schottstaedt (1977), and Chowning (1980). Probably the simplest model of this type to experiment with is the multiple-carrier single-modulator model, where the carrier-to-modulator ratios are integers. This guarantees a harmonic series and it is quite easy to visulize the spectral result, particularly if the carrier frequencies are widely separated and the FM indices are kept small enough to avoid much overlap between the spectra generated by adjacent carriers. Overlaps are actually necessary, however, to provide close fits to original time-varying spectra, so trial-and-error procedures are necessary. Horner *et al* (1993) explored using a *genetic algorithm* as a systematic trial-and-error procedure for finding "best" fixed indices and carrier-to-modulator ratios and the method of least squares to determine amplitude-vs.-time envelopes for each carrier. They found that many acoustic sounds could be closely matched with around 5 carriers. Note that a 6-operator (allowing 3 carriers each with a modulator) FM model was cast in silicon by Yamaha in their 1980's line of FM synthesizers (e.g., the DX7). However, since the amplitude envelopes necessary for close matches (obtained by least squares minimization) are quite complex, having both positive and negative segments, they are beyond the scope of the simple envelope structures provided by the Yamaha synthesizers. However, computer synthesis implementations are not limited by this problem. Figure 5.2.2.6 shows the result for a 3-carrier match to a trumpet tone.

A further problem with the Yamaha implementations of FM synthesis is that they use nonlinear functions to relate "modulator output level" to the FM index. The exact function depends on the particular synthesizer model (Chowning and Bristow, 1986). Thus, it is difficult (but not impossible) for a computer program to simulate Yamaha FM patches configurations or to translate oridinary FM designs into ones that will accurately work on a Yamaha instrument.

We can attempt to predict the efficacy of matching synthesis by using an overall error measure. For example, *average relative error* can be defined as the time-averaged amplitude-normalized rms difference between the synthesized and original spectra :

$$\bar{\varepsilon}_{rel} = (1/\mathrm{DUR}) \int_0^{\mathrm{DUR}} \left[\sum_{k=1}^{N_{hars}} (c_k(t) - c'_k(t))^2 \Big/ \sum_{k=1}^{N_{hars}} c_k^2(t) \right]^{1/2} \quad (5.2.2.12)$$

where $\{c_k(t)\}$ are the orignal acoustic instrument time-varying amplitudes and $\{c'_k(t)\}$ are the matching amplitudes of the multiple-carrier FM instrument. The decrease of average relative error as optimally-matched carriers are added is shown in Figure 5.2.2.7.

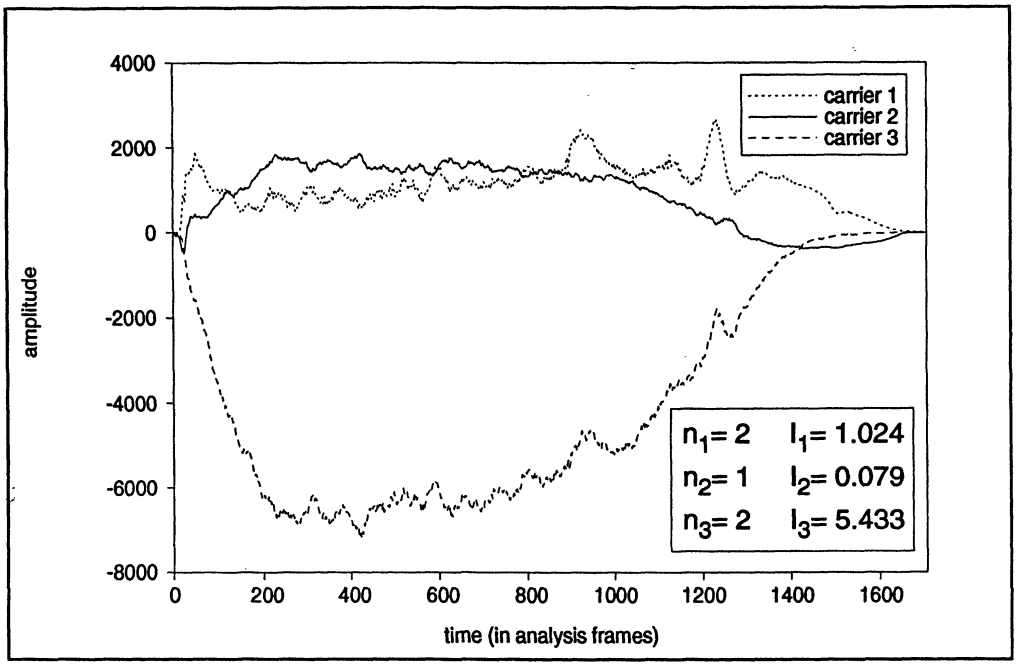


Figure 5.2.2.6 Carrier-to-modulator frequency ratios (n_k), modulator indices (I_k), and carrier amplitude control functions for a three carrier match to a trumpet tone (after Horner *et al*, 1993).

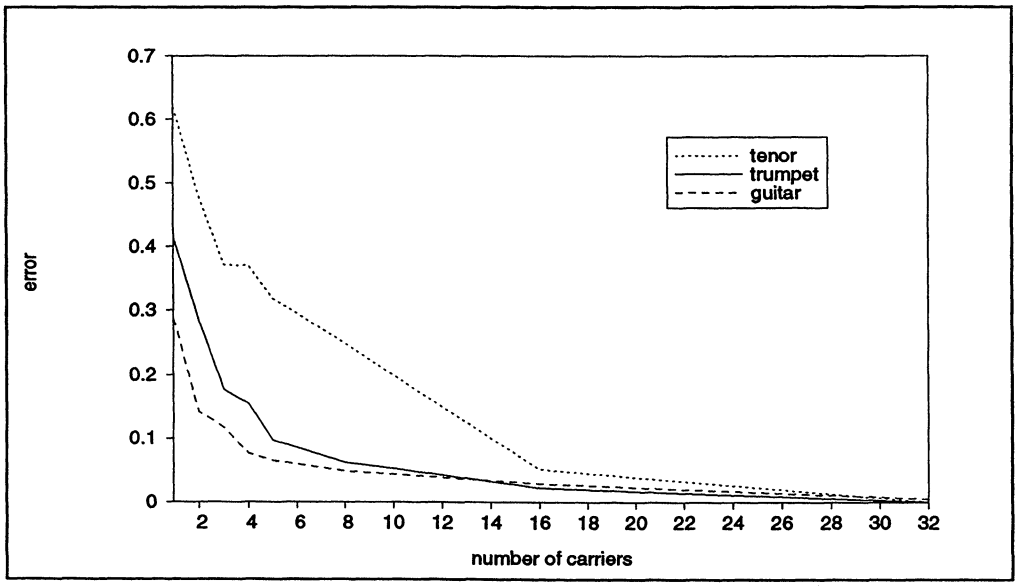


Figure 5.2.2.7 Decrease in relative error between synthetic and original time-varying spectra as more carriers are added for three instrument sounds (after Horner *et al*, 1993).

5.2.2.4 Multiple Modulator FM Synthesis

More than one modulator can be used to drive a single carrier either in parallel or in cascade. The Yamaha DX7, for example, has used several different patches of this type. The method was first discussed by Schottstaedt (1977), and Lebrun (1977) published a mathematical formula for predicting the amplitudes of FM components for the parallel modulator case, which is

$$\sum_{k_1=-\infty}^{\infty}\sum_{k_2=-\infty}^{\infty}\cdots\sum_{k_3=-\infty}^{\infty} J_{k_1}(\alpha_1)\,J_{k_2}(\alpha_2)\ldots J_{k_n}(\alpha_n)\sin(\omega_c t + k_1\omega_{m_1}t + k_1\omega_{m_2}t + \ldots + k_1\omega_{m_n}t) \qquad (5.2.2.12)$$

We have not seen any result for the cascade case. However, despite their analytic complexities, these more complex patches have been used extensively in synthesizers (particularly in the Yamaha line) and useful results have been intuitively-derived.

5.3 Nonlinear Synthesis (Waveshaping)

Since the early days of electronics it has been well known that amplifiers tend to distort sine waves and increase their distortion effects as the sine waves' amplitudes are increased. For high fidelity reproduction with amplifiers, the objective is to reduce the distortion effect, a production of harmonics, to a minimum, usually by supplying passive feedback. With *nonlinear synthesis*, we wish to do something of the opposite: to deliberately create distortion, but of a controlled type, in order create a particular desired spectrum. Also, as the input sine wave's amplitude increases, we expect the strengths of the upper partials to increase at a faster rate, a situation which is typical of the behavior of most wind musical instruments. The central component of nonlinear synthesis is the *nonlinear processor* (sometimes called a *waveshaper*). This is a device which produces an output **y** which is a nonlinear function of an input **x**. The nonlinear processor has no integrators or memory elements; it works "instantaneously". However, the nonlinear processor may be coupled with a filter or amplitude modulator to enhance its production of spectra.

A nonlinear processor is defined by its "transfer fuction", the y-vs.-x function, **F(x)**. Two well known nonlinear processors are "clippers" and "full- wave rectifiers", whose transfer functions are shown below.

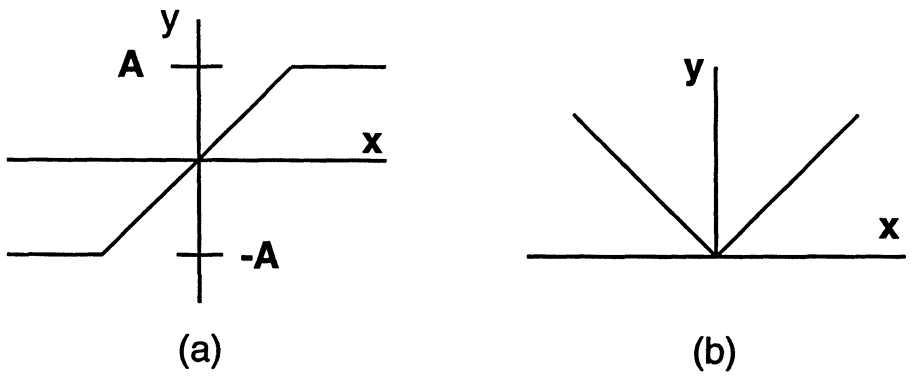


Figure 5.3.1 Simple nonlinear transfer functions.

Usually, the shape of the output waveform depends on the amplitude of an input sine wave. This is true for the transfer function of a) of the above diagram, but not for b). However, the full-wave rectifier is a highly unusual nonlinear function.

Gradual increases in the amplitudes of the harmonics can be obtained with smooth nonlinearities which are defined as polynomials. This is the type of function we will use. First, let

$$y = F(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3 + \ldots + a_n x^n \qquad [5.3.1]$$

Then, let x be the sinusoid

$$x = \alpha \cos(\Theta), \text{ where } \Theta = 2\pi f_1 t \qquad [5.3.2]$$

So we that can write

$$y = \sum_{k=1}^{n} a_k \alpha^k \cos^k(\Theta) \qquad [5.3.3]$$

Since a cosine raised to any power can be expressed as a weighted sum of cosines of certain harmonics, i.e.,

$$\cos^k(\Theta) = (1/2)^{k-1} \sum_{i=0}^{k/2} C_{k,i} \cos((k-2i)\Theta), \qquad [5.3.4a]$$

where $C_{k,i} = k!/((k-i)!i!)$ is the k,i Binomial coefficient, we can see how equation 5.3.3 could be expressed as a sum of harmonics, i.e., $\cos(k\Theta)$ terms.

Note that binomial coefficients can be generated using the the recursion formula

$$C_{k,i} = C_{k-1,i-1} + C_{k-1,i} \qquad [5.3.4b]$$

Equation 5.3.4a contains odd or even harmonic terms according as k is odd or even. The highest harmonic in equation 5.3.4a is harmonic k. Here are a couple of examples:

$$\cos^2(\Theta) = .5 + .5\cos(2\Theta)$$

$$\cos^3(\Theta) = .75\cos(\Theta) + .25\cos(\Theta)$$

We can then substitute equation 5.3.4 into equation 5.3.3, and after a fair amount of rearranging, we arrive at

$$y = \sum_{k=1}^{n} d_k(\alpha) \cos(k\Theta) \qquad [5.3.5a]$$

where

$$d_k(\alpha) = 2 \sum_{i=k,2}^{i \le n} p_{k,i} (\alpha/2)^i a_i \qquad [5.3.5b]$$

and

$$p_{k,i} = i!/(((i-k)/2)!((i+k)/2)!) \qquad [5.3.5c]$$

We have emphasized that d_k is a function of α because we are interested in how the harmonic amplitudes vary with the amplitude of the input sine wave. As an analogy to the FM case, we can refer to α as the *index* of nonlinear processing.

To summarize, given the coefficients of a nonlinear polynomial $\{a_k\}$ and the amplitude of an input sine wave (α), the output harmonic amplitudes $\{d_k\}$ which result from distortion may be calculated. The polynomial F(x) defines the "transfer function" of the nonlinear processor F(x). The process may be depicted as shown in Figure 5.3.2

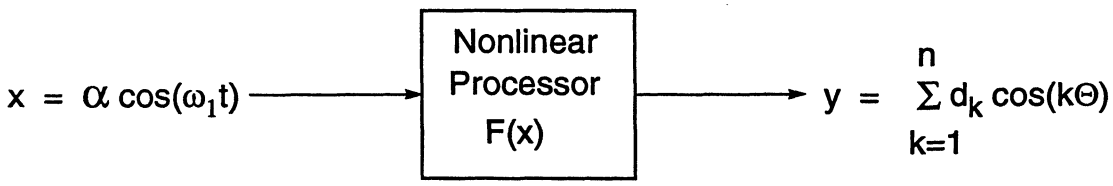


Figure 5.3.2 The Nonlinear Processor Synthesizer

Our next objective is to find the best polynomial to generate a particular "target spectrum" $\{d_{k_0}\}$. This will be the spectrum which is produced when the input sine wave has the target amplitude $\alpha_{0,}$, which we normally take to be unity. Starting with the harmonic series expansion of the output

$$y = F(x) = F(\alpha_0 \cos(\Theta)) = \sum_{k=0}^{n} d_{k_0} \cos(k\Theta) \qquad [5.3.6a]$$

we note that from equation 5.3.2 we can express $\Theta = \mathbf{acos}(x/\alpha_0)$ [5.3.6b]

which leads to $$y = \sum_{k=0}^{n} d_{k_0} \cos(k \,\mathbf{acos}(x/\alpha_0)). \qquad [5.3.6c]$$

$\cos(k\,\mathbf{acos}(x/\alpha_0))$ can be recognized as the Tchebycheff polynomial $T_k(x/\alpha_0)$. Some Tchebycheff polynomials are given below:

$$T_0 = 1, \quad T_1 = x/\alpha_{0,} \quad T_2 = (x/\alpha_0)^2 - 1, \quad T_3 = 4(x/\alpha_0)^3 - 3(x/\alpha_0), \qquad [5.3.7a]$$
$$T_4 = 8(x/\alpha_0)^4 - 8(x/\alpha_0)^2 + 1, \quad T_5 = 16(x/\alpha_0)^5 - 20(x/\alpha_0)^3 + 5(x/\alpha_0).$$

Note that odd-ordered Tchebycheff polynomials contain only odd-power terms, and even-ordered Tchebycheff polynomials contain only even-power terms. They can be successively derived by means of the recursion formula

$$T_{k+1}(x/\alpha_0) = 2(x/\alpha_0)\, T_k(x/\alpha_0) - T_{k-1}(x/\alpha_0) \qquad [5.3.7b]$$

We can see from these formulas that odd (even) polynomials are always generated by odd (even) harmonic terms. Also, n is the highest power which results from equation 5.3.6c.

By plugging the Tchebycheff polynomials into Equation 5.3.6c and collecting terms, a power series is arrived at with coefficients involving combinations of the d_{k_0}. These terms can then be equated to

respective a_k's. After much manipulation the following general formula can be derived:

$$a_i = .5(2/\alpha_o)^k \sum_{k=i,2}^{k\le n} q_{i,k}\, d_{k_o} \qquad [5.3.8a]$$

where $$q_{i,k} = (-1)^{(k-i)/2}\, k((k+i-2)/2)!/(i!((k-i)/2)!) \qquad [5.3.8b]$$

Therefore, if it is desired to synthesize a spectrum for a particular value of α_o (usually unity), equation 5.3.8 gives the polynomial coefficients which will do the job. Then, equations 5.3.5b,c predict the resulting harmonic amplitudes for any value of α, which normally will be varied during synthesis of a sound. For a certain subset of possible $\{d_k\}$ target spectra, the individual harmonic amplitudes will increase monotonically as α increases, making it easy to control the spectrum's brightness. Experience has shown that target spectra which roll off suitably fast with increasing harmonic number will exihibit this monotonic property, but it is an open question as to what exact rolloff relationship is required.

However, an important feature of nonlinear processes is exhibited at *low amplitudes* (small values of α). In this case, equation 5.3.5b reduces to

$$d_k = 2\, a_k\, (\alpha/2)^k \qquad [5.3.9a]$$

with the special case

$$d_1 = a_1\, \alpha\,. \qquad [5.3.9b]$$

Combining these two equations yields the relationship

$$d_k = [2\, a_k/(2\, a_1)^k]\, d_1^{\,k} \qquad [5.3.10a]$$

Stated in words, this means that the kth harmonic amplitude varies as the kth power of the first harmonic amplitude. This can also be expressed in decibel units:

$$D_k = k\, D_1 + D_{o_k} \qquad \text{(decibels)} \qquad [5.3.10b]$$

where $D_k = 20 \log_{10}(d_k)$ and $D_{o_k} = 20 \log_{10}(2\, a_k/(2\, a_1)^k)$ [5.3.10c]

In other words, **the level (in decibels) of the kth harmonic plotted vs. the level of the first harmonic is a straight line with slope equal to k.**

Figure 5.3.3 shows the D_k which result from the polynomial $y = x + x^2 + x^3 + x^4 + x^5 + x^6 + x^7 + x^8$ for harmonics 2 to 8 versus harmonic 1. While the low level behavior follows the predicted slope, note that some of the curves deviate from straight line behavior as α increases.

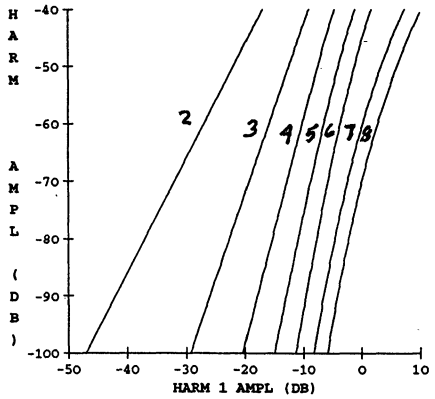


Figure 5.3.3 Typical growth of harmonic amplitude levels as α increases.

5.3.1 **Computation of polynomial coefficients and harmonic amplitudes.** To systematize the computations implied by Equations 5.3.5b,c and 5.3.8a,b we can use matrix algebra. Let $\{d_k\}$ and $\{a_i\}$ be column matrices with indices running from 0 to n. We can then relate $\{d_k\}$ and $\{a_k\}$ by means of the n+1 by n+1 matrices **P** and **Q** as follows:

$$[d_k] = 2\,P\,[a_i\,(\alpha/2)^i] \quad \text{and} \quad [a_i] = .5\,\{(2/\alpha)^i I\}\,Q\,[d_k]$$

where

$$P = \begin{array}{c} i=0 \\ k=0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \\ 9 \end{array} \begin{array}{c} 1 \\ \begin{pmatrix} 1 & 0 & 2 & 0 & 6 & 0 & 20 & 0 & 70 & 0 \\ & 1 & 0 & 3 & 0 & 10 & 0 & 35 & 0 & 126 \\ & & 1 & 0 & 4 & 0 & 15 & 0 & 56 & 0 \\ & & & 1 & 0 & 5 & 0 & 21 & 0 & 84 \\ & & & & 1 & 0 & 6 & 0 & 28 & 0 \\ & & & & & 1 & 0 & 7 & 0 & 36 \\ & & & & & & 1 & 0 & 8 & 0 \\ & & & & & & & 1 & 0 & 9 \\ & & & & & & & & 1 & 0 \\ & & & & & & & & & 1 \end{pmatrix} \end{array} \begin{array}{c} 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \\ 9 \end{array}$$

and

$$Q =$$

i \ k	0	1	2	3	4	5	6	7	8	9	
0	1	0	-2	0	2	0	-2	0	2	0	0
1		1	0	-3	0	5	0	-7	0	9	1
2			1	0	-4	0	9	0	-16	0	2
3				1	0	-5	0	14	0	-30	3
4					1	0	-6	0	20	0	4
5						1	0	-7	0	27	5
6							1	0	-8	0	6
7								1	0	-9	7
8									1	0	8
9										1	9

The matrix elements for both P and Q are given by equations 5.3.5c and 5.3.8b. They can also be determined by simple recursion formulas. The recursion formula for $p_{k,i}$ is

$$p_{k,i} = p_{k-1,i-1} + p_{k+1,i-1}, \; k > i; \text{ also, } p_{k,k} = 1 \text{ and } p_{0,i} = 2p_{1,i-1} \qquad (5.3.11a)$$

For the matrix Q the first line is as follows:

1 0 -2 0 2 0 -2 0 2

The other lines can be computed according to the recursion formula

$$q_{i,k} = q_{i-1,k-1} - q_{i,k-2} \qquad (5.3.11b)$$

Examples:

Given that $F(x) = x + 2x^2 + 3x^3$, calculate the d_k's for $\alpha = 0.6$ and $k = 1, 2, 3$:

$$\begin{pmatrix} d_1 \\ d_2 \\ d_3 \end{pmatrix} = 2\times \begin{pmatrix} 1 & 0 & 3 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \times \begin{pmatrix} 1\times 0.3 \\ 2\times 0.09 \\ 3\times 0.027 \end{pmatrix} = \begin{pmatrix} 1.086 \\ 0.36 \\ 0.162 \end{pmatrix} \qquad (5.3.12a)$$

Find the a_i's which will yield the spectrum $d_{k_o} = 11,12,13$ (for $k = 1,2,3$) for $\alpha_o = 0.5$:

$$\begin{pmatrix} a_1 \\ a_2 \\ a_3 \end{pmatrix} = \frac{1}{2}\times \begin{pmatrix} 4 \\ 16 \\ 64 \end{pmatrix} \begin{pmatrix} 1 & 0 & -3 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \times \begin{pmatrix} 11 \\ 12 \\ 13 \end{pmatrix} = \begin{pmatrix} -56 \\ 96 \\ 416 \end{pmatrix} \qquad [5.3.12b]$$

In summary, the coefficients of a nonlinear polynomial F(x) can be derived for an arbitrary target spectrum $\{d_{k_0}\}$ using Equation 5.3.8, where $\alpha_0 = 1$ can be used without loss of generality. The spectrum $\{d_k\}$ that is generated as α varies (with time as obtained from an envelope generator) can be derived using Equation 5.3.5b. The nonlinearity can be characterized in terms of the spectrum it produces by plotting d_k vs. d_1 or, in decibel units, the equivalent D_k vs. D_1, with α as an implicit parameter of these curves. For low amplitudes, equation 5.3.10 can be used to predict the behavior of these curves. The spectrum produced by the nonlinear processor tends towards a pure sine wave for low values of α and, in general, has increasing spectral centroid or brightness as α increases.

5.3.2 Matching Nonlinear Synthesis to Acoustic Instrument Sounds

The nonlinear method can be applied to the problem of synthesizing tones which match the time-varying harmonic spectra of musical instrument tones . Time-varying spectra for cornet and alto saxophone tones (played ***mf***) are shown in Figure 5.3.4. Note in both cases that the higher-numbered partials tend to increase in amplitude later during the attack and decrease earlier during the decay than the lower-numbered partials. As a consequence, the "brightness" (BR) or spectral centroid increases monotonically with the overall (RMS) amplitude. (Recall that BR and RMS are defined by equations 5.0.1 and 5.0.2.)

BR vs. time is graphed for the two musical instrument tones in Figure 5.3.5. We see that BR changes quite rapidly during the attack and decay, and that for the saxophone a short burst of relatively high BR value occurs during the attack phase. If this burst had occurred during high RMS amplitude, it would be perceptually very obvious; since it actually occurs during a low amplitude phase, it is a rather subtle feature of the attack. The corresponding RMS envelopes are shown in Figure 5.3.6; they parallel the activity of the BR curves, but are quite different. BR is plotted as a function of RMS in Figure 5.3.7. We see that the curves are approximately one-to-one and monotonically increasing except for deviations during the attack phases of the tones. For the cornet, the slope of the BR vs. RMS curve is quite steep for lower amplitudes and more shallow for values of RMS greater than 40. We see a similar effect for the saxophone. Comparisons of these results for ***mf*** tones with those of ***pp*** and ***ff*** tones bear out the mononicity of these relationships.

The importance of the spectral centroid measurement is due to its close relationship with the perceptual concept of brightness. Listening tests have shown that next to loudness, pitch, and duration, brightness may be the most important perceptual factor for sounds in general. For nonlinear synthesis, the importance of the BR vs. RMS relationship for a musical instrument played at a given pitch lies in the fact that nonlinear processors also generally yield monotonically increasing BR vs. RMS relationships. Adding a high pass filter to the processor's output, provides an extension of the range of BR while retaining the monotonic property. In fact, this extended range is needed to match the BR range of most acoustic instruments. Moreover, the high pass filter provides a strong analogy to the acoustics of wind instruments since the frequency response of a wind musical instrument is generally some kind of high pass filter.

Arranging for the nonlinear synthesis of a tone to match the time-varying brightness of an acoustic sound generally reduces to five steps: 1) Measure BR(t) for the acoustic sound. 2) Measure the BR vs. index characteristic for the nonlinear process. 3) Assuming that the BR vs. index characteristic is monotonically increasing, invert this characteristic to produce the corresponding index vs. BR characteristic. 4) Use the measured BR(t) to produce the data for index vs. time ($\alpha(t)$). 5) Use $\alpha(t)$ as the amplitude of the sine wave input to the nonlinear process in order to generate the output signal. Or use a predictive model of the nonlinear process to generate the time-varying spectrum of the output.

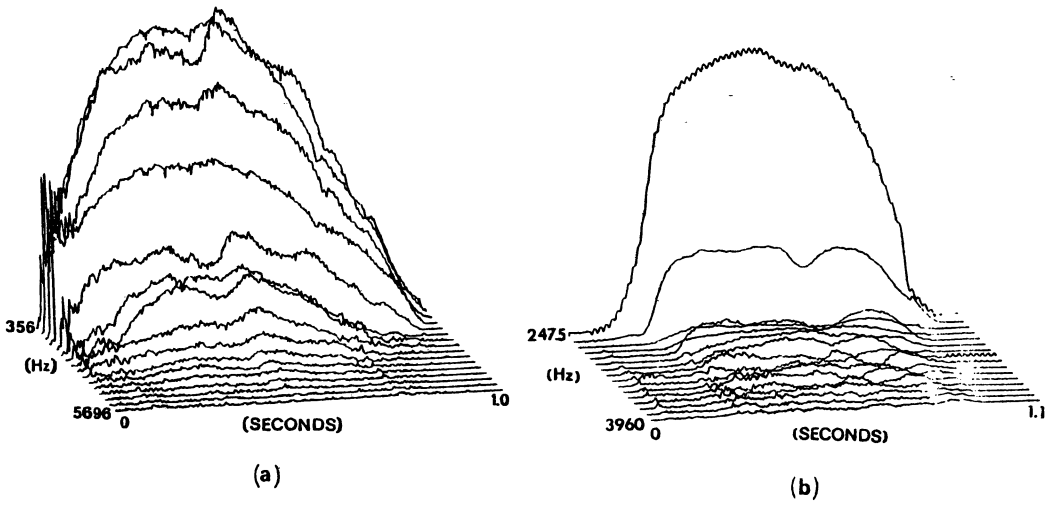


Figure 5.3.4 Time-variant spectrum analysis data for (a) a cornet tone (356 Hz, mf); (b) an alto saxophone tone (247.5 Hz, mf). Amplitude is vertical. (After Beauchamp, 1982).

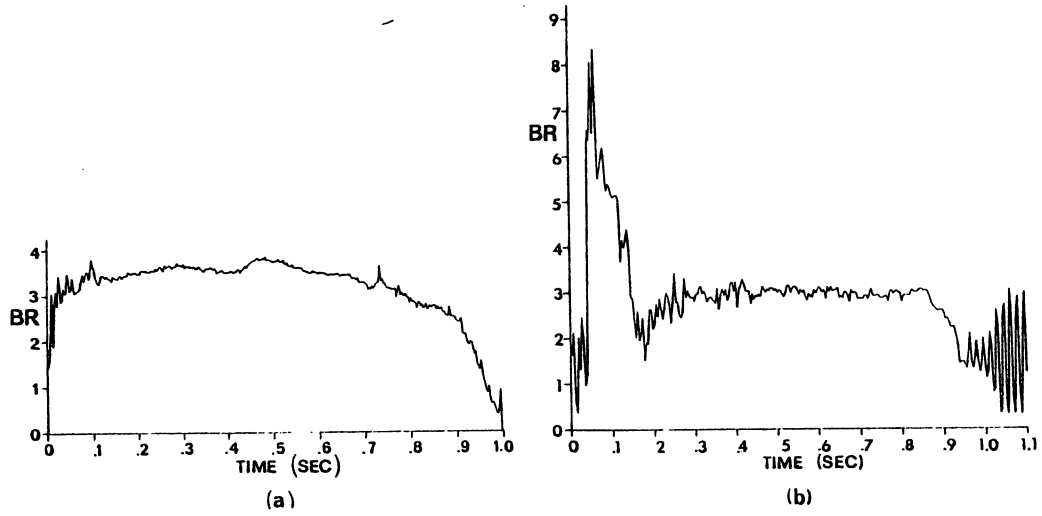
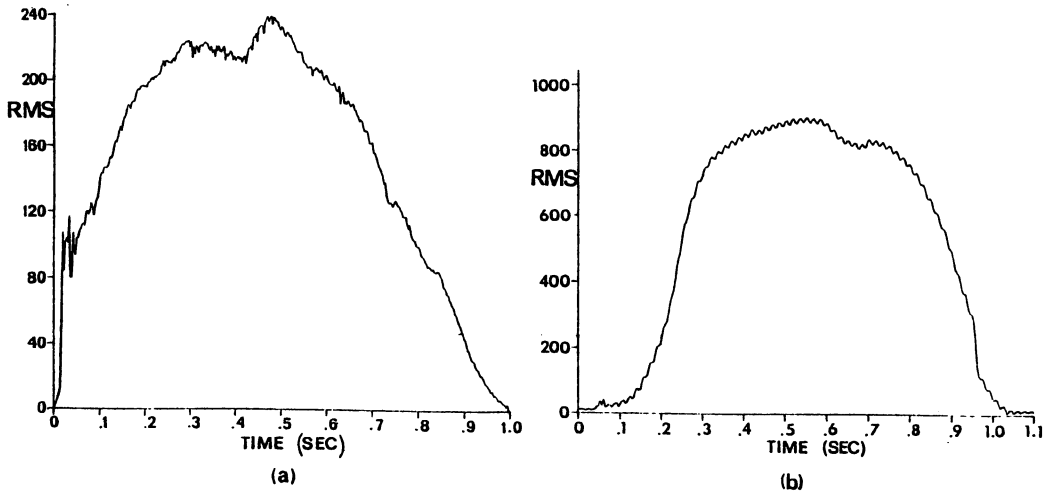


Figure 5.3.5 BR vs. time for (a) the cornet tone; (b) the alto saxophone tone. (After Beauchamp, 1982).



Figre 5.3.6 RMS amplitude vs. time for (a) cornet; (b) alto saxophone. (After Beauchamp, 1982).

Even though a synthesis method may match the brightness vs. time characteristic of an instrument sound, the resulting BR vs. RMS characteristic of the nonlinear processor may not match the sound. However, by using a post-multiplier ($\beta(t)$) at the output of the synthesis model, it can be forced to match this characteristic. Alternatively, $\beta(t)$ can be used to match the entire spectrum in a least-squares sense, and this is the approach that we adopt here.

5.3.2.1 **BR vs. INDEX properties of nonlinear processes**

Increasing the index (α, the amplitude of the incoming sine wave) of a nonlinear process generally has the effect of increasing the output brightness, although in some cases it could actually go down. In order to compare how this works for three different nonlinear systems, we can look at typical BR vs. α characteristics for these systems as well as the corresponding growth behavior of some individual harmonic amplitudes (c_k's).

If we consider simple two-operator FM with equal carrier and modulation frequencies, we see that there is no limit on the range of BR as the index is increased, but the curve is not strictly monotone increasing (see Figure 5.3.8a) ; i.e., certain BR values can be produced with two different index values. Moreover, letting the amplitude of the FM signal be proportional to the index, we see from Figure 5.3.9a that the harmonic amplitudes are not monotone increasing, but behave in a complex way depending on the Bessel functions. That we can be induced to hear a "brass-like sound" from this model must be due to certain sloppinesses in our timbre perception mechanisms. However, it should be clear that with this system we can never accurately match the spectrum of a real brass tone.

Nevertheless, if we approximate the FM BR-vs.-α relationship by a *straight line*,

$$BR \approx 0.56\,\alpha + 1 \qquad (5.3.13)$$

we can use it to match the brightness of a real tone. Inverting this formula gives us a simple way to calculate the approximate α for any desired BR. Even so, when we synthesize tones using this technique, rather inferior results, worse than intuitively-derived ones, result.

The second system to consider is a polynomial nonlinear processor, defined by y=F(x), without a filter after it (i.e., the case we considered in the previous section). If we match a particular musical instrument spectrum, say, a *cornet mf* spectrum, for α=1, we can vary α from 0 to 1 and beyond and plot the BR of the resulting spectrum vs. the index. Cornet "reference spectra" for three dynamics are given in Table 5.2.1. The polynomial coefficients for F(x) can be calculated using Equation 5.3.8, and the nonlinear processor output spectrum for any value of α can be calculated using Equation 5.3.5b,c. The spectrum-vs.-α data can then be used to calculate the BR vs. α curves shown in Figure 5.3.8b. (For calculation of BR, we use Equation 5.0.1a,b, where "c_k" is replaced by "d_k".) The *mf* curve is not very useful because the BR value needed to match the *ff* spectrum cannot be attained. The curve based on the *ff* reference spectrum has a chance, but it is decidedly not monotonic. Moreover, as shown in Figure 5.3.9b, the c_k's generated by this method do not themselves increase monotonically.

The third system is a nonlinear processor followed by a high pass filter. The filter chosen is a second order high pass type of the form

$$H(s) = (s/\omega_c)^2/\{(s/\omega_c)^2 + 2\zeta(s/\omega_c) + 1\} \qquad (5.3.14)$$

Values of ζ (the damping factor) and f_c (the cutoff frequency) are chosen for a best match of the

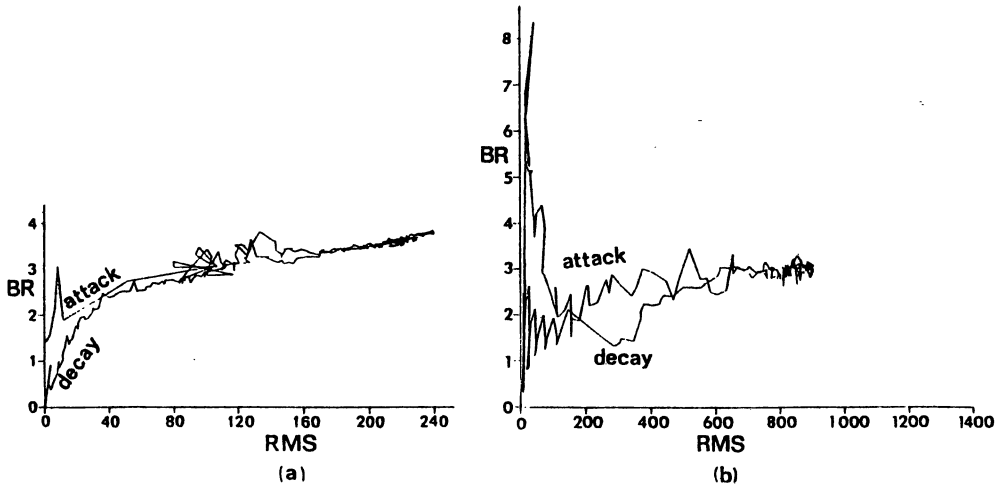


Figure 5.3.7 BR vs. RMS amplitude for (a) cornet (b) alto saxophone (After Beauchamp, 1982).

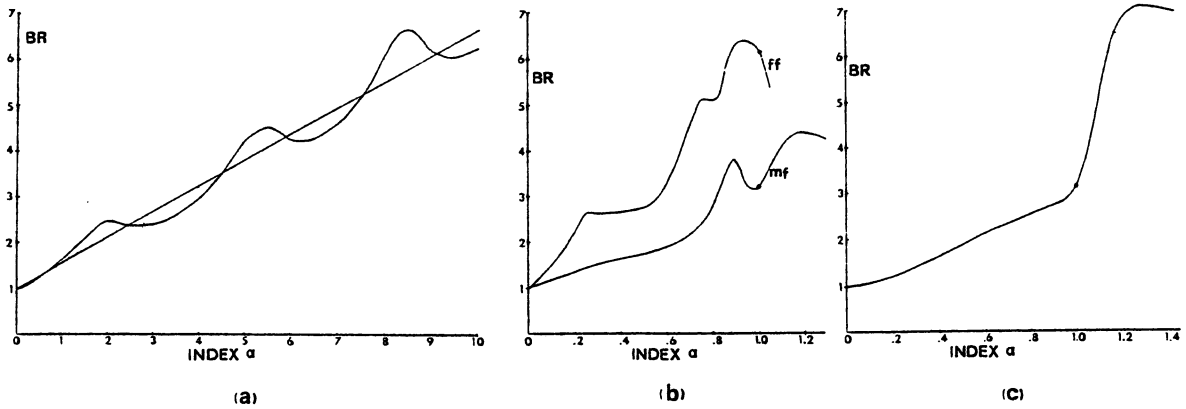


Figure 5.3.8 BR vs. index (α) for three synthesis methods: (a) Standard 2-operator frequency modulation. (b) Nonlinear processing with no post-filter. (Curves for two cornet reference spectra are shown (*mf* and *ff*).) (c) Nonlinear synthesis with second-order high pass post-filter (mf cornet reference spectrum). (After Beauchamp, 1982).

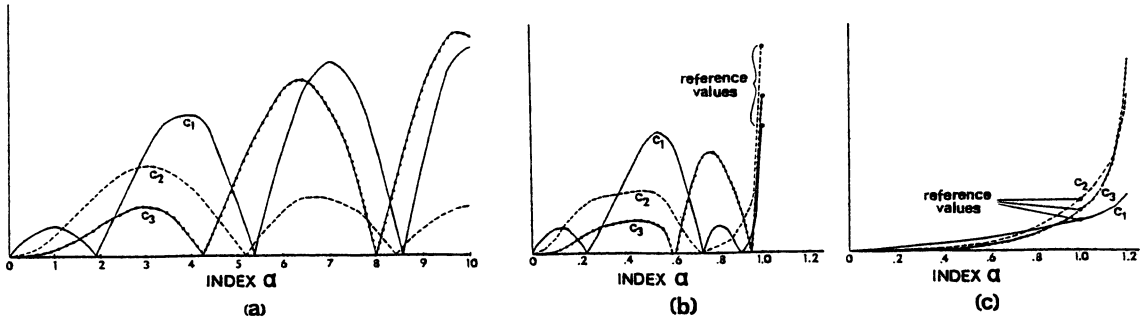


Figure 5.3.9 Amplitude buildup of first three harmonics. (a) Frequency modulation. (b) Nonlinear synthesis with no filter (*ff* cornet reference spectrum). (c) Nonlinear synthesis with high pass post-filter (*mf* cornet reference spectrum). (After Beauchamp, 1982).

acoustical data, either by direct measurement of the instrument's high pass characteristic or by optimizing the overall results. The values we used for the cornet and alto saxophone are given in Table 5.3.1. Graphs of $|H(j2\pi f)|$ vs. f for the filter parameter values chosen are shown in Figure 5.3.10.

Dividing the output reference spectrum (the c_{k_0}'s) by the filter frequency response sampled at harmonics of the fundamental frequency f_1 (i. e., $|H(j2\pi kf_1)|$), yields the nonlinear processor output reference spectrum (the d_{k_0}'s). (This corresponds to the signal spectrum in a wind instrument's mouthpiece.) This derived d_{k_0} reference spectrum is then used to generate the distorting function F(x). Graphs of F(x) vs. x for the cornet and alto sax cases are shown in Figure 5.3.11.

By varying the amplitude α of the sine wave going into F(x), we can generate a signal whose harmonic amplitudes are the d_k's , which correspond tol the reference d_{k_0} spectrum when $\alpha = 1$. But these d_k's are then modified by the filter which, according to its characteristic, converts them into the c_k's. The variation of the first three c_k's as a function of the index α is shown in Figure 5.3.8c. In this case, we see that the behavior is strictly monotonic for each c_k! Moreover, the BR-vs.-α curve shown in Figure 5.3.7c is also strictly monotonic (until a maximum is reached at α=1.28), and considering that the processor function F(x) was based on ***mf*** reference spectrum, we see that , quite remarkably, the range of BR is sufficient to accommodate an ***ff*** spectrum and beyond. Therefore, of the three systems, the nonlinear processor/high pass filter (NLF) approach seems to have the best chance for success in matching the brightness and total spectrum for the time-variant situation.

5.3.2.2 Synthesis and matching technique with the NLF approach

Once the processing function F(x) and the high pass filter function H(s) have been chosen, it remains to find appropriate time-varying functions $\alpha(t)$ and $\beta(t)$ (the post-multiplier envelope) to match the BR-vs.-time and spectrum-vs.-time characteristics of an acoustic tone. Three block diagrams for possible synthesizers are shown in Figure 5.3.12, but let us for the moment focus on the one given by Figure 5.3.12a. Here we have separate envelope generators controlling the $\alpha(t)$ and $\beta(t)$ functions. (The block diagram terminology is that used in the computer music literature: OSCILI is an interpolating generator with (in this case) a sine waveform; its left input controls the amplitude and its right input controls the frequency of its output; VFMULT uses a nonlinear processor using F(x); its left input controls its output amplitude and its right input controls the value of x.)

As discussed above, the envelope, $\alpha(t)$, is determined by first determining the BR-vs.-α characteristic for the processor/filter combination, then inverting this function so that a value of BR automatically determines a value of α, and finally, by using the graphs of BR-vs.-time given in Figure 3a, b to automatically generate graphs of α vs. time -- graphs which are correct for generating the requisite BR-vs.-time characteristics.

Note that in the synthesis models shown in Figure 5.12b,c , BR"(t) is used to control the index of the nonlinear process. The modified BR"(t) is a linear function of BR(t) and is designed to always vary between 0 to 10. It is given by

$$BR'' = 10\,(BR - 1)/BR_{maxp} \qquad (5.3.15)$$

where BR_{maxp} is the maximum value of BR producible by the synthesis process (see Fig. 5.3.8).

Reference Spectra ($c_{1_0}, c_{2_0}, c_{3_0}, \ldots, c_{n_0}$)

Cornet (*mf*)	65.8., 110., 84.5, 103.4, 27.4, 14.1, 66.1, 4.3, 0.6, 1.7, 1.5, 1.1, 0.8, 0.5, 0.5, 0.2
Cornet (*ff*)	276., 448., 340., 387., 152., 189., 227., 245., 331., 114., 123., 86., 55., 48., 52., 42.
Alto Sax (*mf*)	754.9, 237.9, 37.8, 36.2, 81.3, 49.7, 44.6, 27.3, 25.2, 22.4, 21.7, 13.9, 16.1, 14.1, 10.4.,7.7

Other Data

	f_a	f_c	ζ	BR_{maxp}	xmax
Cornet	356	3205	0.442	7.1	1.28
Alto Saxophone	247.5	2781	0.229	8.1	1.21

Table 5.3.1 Nonlinear analysis/synthesis data for cornet and alto saxophone tones. (After Beauchamp, 1982)

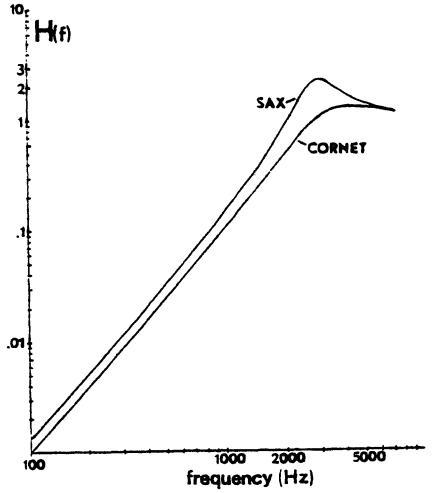


Figure 5.3.10 High pass filter response functions used for nonlinear synthesis of the cornet and the alto saxophone. (After Beauchamp, 1982).

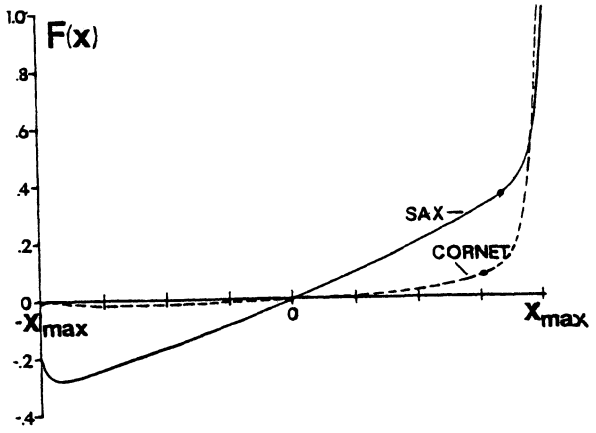


Figure 5.3.11 Nonlinear processing functions used for synthesis of the cornet and alto saxophone. Functions have been normalized so tht $F(x_{max}) = 1$. Closed circles (dots) on these curves indicate the values of F for x =1, which in each case represents the sine wave amplitude necessary to produce the reference spectrum. (After Beauchamp, 1982).

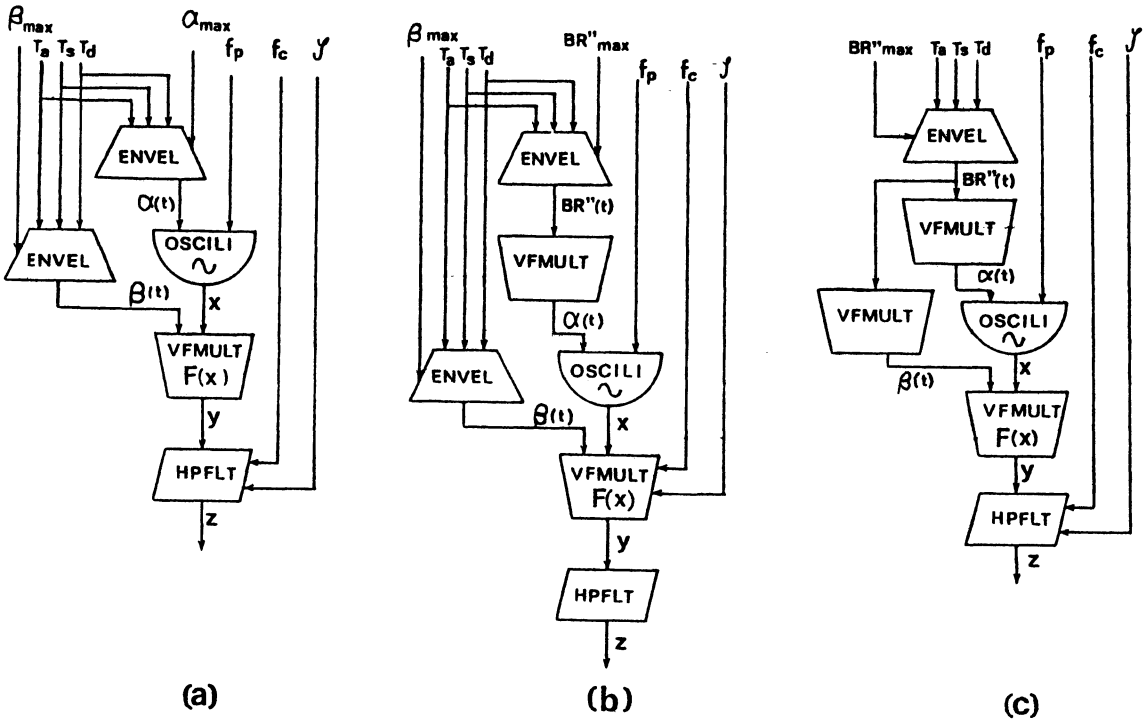


Figure 5.3.12 Flow diagrams for nonlinear/filter synthesis models. (a) Direct control by α(t) and β(t) envelopes. (b) Translated control by BR'' and direct control by β(t) envelopes. (c) Translated control by BR'', where β is a function of BR''. (After Beauchamp, 1982).

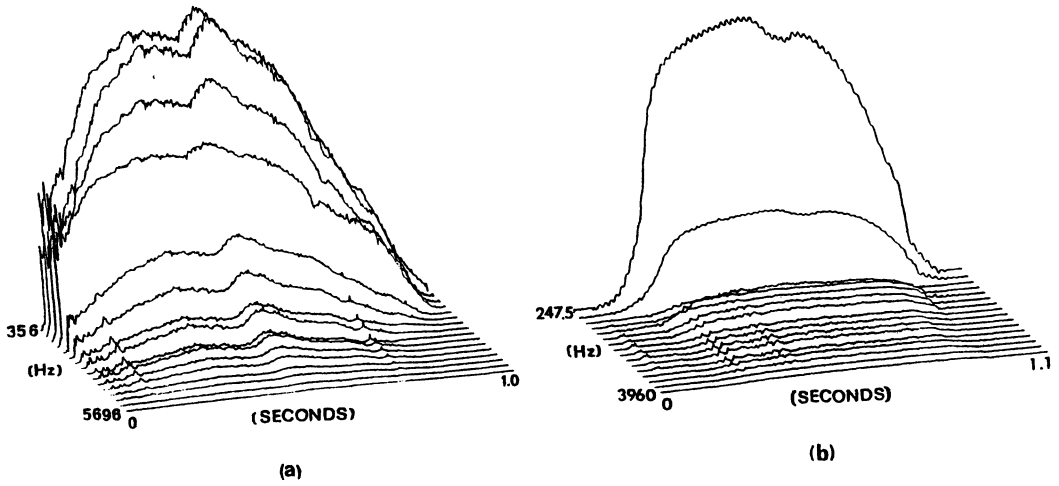


Figure 5.3.13 Simulated time-variant spectrum analysis data for nonlinear/filter synthesis of (a) the cornet tone ; (b) the alto saxophone tone. These graphs should be compared to the original data shown in Figure 5.3.4. (After Beauchamp, 1982).

BR" is preferable to α for control because BR" is much more directly related to the "brightness" of the sound being synthesized.

Next, $\beta(t)$, the post-multiplier, is determined in order to match the filter's output spectrum to that of the instrument tone for every instant of time. β is a variable gain control, and all it can do is increase or decrease the filter's output amplitude. However, for every time instant, we can arrange for β to produce the spectrum which most closely matches the original in the least-squared error sense. If the NLF time-varying spectrum components are given by

$$c_k"(t) = \beta(t)\, c_k'(t)\} \,. \qquad (5.3.16)$$

and the corresponding components of the origininal sound are given by $\{c_k\}$, it turns out that the optimum time-varying $\beta(t)$ can be calculated as

$$\beta(t) = \Sigma\, c_k(t)\, c_k'(t) \,/\, \Sigma\, c_k'^{\,2}(t) \,. \qquad (5.3.17)$$

Alternatively, the original rms amplitude could be matched using

$$\beta(t) = \mathrm{sqrt}(\, \Sigma\, c_k^2(t) \,/\, \Sigma\, c_k'^{\,2}(t)\,) \,. \qquad (5.3.18)$$

Time-varying synthetic spectra $\{c_k"(t)\}$ resulting from this technique are shown in Figures 5.3.13a, b and should be compared to Figures 5.3.4a ,b as a way of gauging the accuracy of synthesis.

β may have a strong dependence on BR, as is shown in Figure 5.3.14a & b. The synthesis method depicted in Figure 5.3.12c utilizes this dependence to eliminate the need for a separate post-multiplier envelope. BR" (a version of BR which is scaled to a 0 to 10 range) is automatically translated into β according to the derived β vs. BR relationship. However, this model ignores the fact that β and BR may be independent during the attack transient and so has not proven to be as satisfactory as the model which employs independent β and. BR time variations.

A summary of the relative errors (see Eq. 5.2.2.12) measured between the synthesized and the original acoustic tones discussed above are given in Table 5.3.2. Table 5.3.3 gives hand-fit straight line coordinate data for $\alpha(t)$ and $\beta(t)$ envelopes for the cornet and alto sax tones, and Table 5.3.4 gives the BR" vs. time envelopes and the α vs. BR" and β vs. BR" translation functions needed for the synthesizers of Figures 5.3.12b & c.

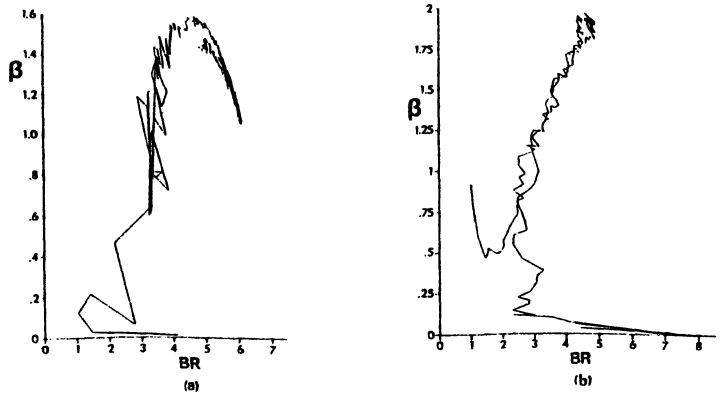


Figure 5.3.14 β (post-multiplier) vs. BR for (a) a 356 Hz cornet tone (*ff*); (b) an alto saxophone tone (*ff*). (After Beauchamp, 1982).

	Cornet (356 Hz)		Alto Saxophone (247.5 Hz)	
	Nonlinear/Filter	FM	Nonlinear/Filter	FM
pp	0.147	0.440	0.089	0.511
mf	0.088	0.565	0.039	0.357
ff	0.264	0.719	0.108	0.559

Table 5.3.2 Relative errors measured between synthesized and orginal cornet and alto saxophone tones, based on time-variant spectra, for two instruments, three dynamics, and two synthesis techniques. (After Beauchamp, 1982).

	Cornet (*mf*, 356 Hz):
t, α	0., 0.5. 0.01, 0.95 0.013,0.64 0.017, 0.95 0.025, 1.03 0.03, 0.97 0.05, 1.03 0.06, 1.00 0.10, 1.05 0.11, 1.03 0.285, 1.05 0.43, 1.04 0.47, 1.05 0.71, 1.01 0.73, 1.03 0.86,0.93 0.91, 0.82 1.01, 0.
t, β	0., 0. 0.01, 0.11 0.013, 0.06 0.017, 0.77 0.025, 0.63 0.03, 0.93 0.05, 0.62 0.06, 0.84 0.10, 0.77 0.11, 1.00 0.16, 1.20 0.285, 1.35 0.43, 1.37 0.47, 1.40 0.56, 1.30 0.58, 1.32 0.71, 1.10 0.73, 0.83 0.75, 0.91 0.86, 0.73 0.98, 0.63 1.01, 0.
	Alto Saxophone (*mf*, 247.5 Hz):
t, α	0., 0. 0.06, 1.21 0.07, 1.05 0.14, 1.025 0.175, 0.93 0.19, 0.985 0.82, 1.00 1.10, 0.
t, β	0., 0. 0.04, 0.02 0.06, 0.01 0.11, 0.03 0.15, 0.07 0.22, 0.27 0.27, 0.64 0.32, 0.78 0.40, 0.84 0.57, 0.90 0.67, 0.80 0.71, 0.83 0.76, 0.80 0.82, 0.73 0.88, 0.55 0.925, 0.39 0.94, 0.47 0.96, 0.15 1.01, 0.

Table 5.3.3 Hand-fit piecewise linear approximations for α(t) and β(t) time envelopes for *mf* cornet and alto saxophone tones. (After Beauchamp, 1982).

	Cornet (*mf*, 356 Hz):
t, BR"	0., 1.5 0.01, 3.1 0.013, 2.1 0.017, 3.1 0.025, 4.2 0.03, 3.3 0.05, 4.2 0.06, 3.6 0.10, 4.9 0.11, 4.2 0.285, 4.9 0.43, 4.6 0.47, 4.9 0.71, 3.8 0.73, 4.2 0.86, 3.1 0.91, 2.7 1.01, 0.
BR", α	0., 0. 0.5, 0.24 3.3, 0.98 4.9, 1.05 7.4, 1.11 9.5, 1.19 10.0, 1.28
BR", β	0., 0. 4.0, 1.6 5.4, 1.9 10.0, 2.0
	Alto Saxophone (*mf*, 247.5 Hz):
t, BR"	0., 0. 0.06, 10.0 0.07, 6.2 0.14, 4.5 0.175, 1.4 0.19, 2.2 0.82, 2.9 0.95, 0.6 1.10, 0.
BR", α	0., 0. 0.2, 0.4 0.6, 0.7 1.0, 0.85 1.5, 0.95 2.9, 1.0 8.4, 1.09 9.4, 1.12 9.8, 1.15 10.0, 1.21
BR", β	0., 0. 4.0, 1.6 5.4, 1.9 10.0, 2.0

Table 5.3.4 Brightness envelopes and α vs. BR" and β vs. BR" translation functions needed for nonlinear/filter synthesis of cornet and alto saxophone tones. (After Beauchamp, 1982)

References on Dynamic Spectrum Synthesis

1. M.E. Clark, D.A. Luce, R. Abrams, H. Schlossberg, and J. Rome, "Preliminary Experiments on the Aural Significance of Parts of Tones of Orchestral Instruments and on Choral Tones", *J. Audio Engr. Soc.*, Vol. 11, pp. 45-54 (1963).

2. E.L. Saldanha and J.F. Corso, "Timbre Cues and the Identification of Musical Instruments", *J. Acoust. Soc. Am.*, Vol. 36, pp. 2021-2026 (1964).

3. Robert A. Moog, "A Voltage-Controlled Low-Pass HIgh-Pass Filter for Audio Signal Processing", *Audio Engr. Soc. Preprint No. 413* (1965).

4. D.A. Luce and M.E. Clark, "Physical Correlates of Brass-Instrument Tones", *J. Acoust. Soc. Am.*, Vol. 42, pp 1232-1243 (1967).

5. W. Carlos and B. Folkman, "Switched on Bach", *Columbia Records*, MS7194 (1968).

6. B.S. Atal and S. L. Hanauer, "Speech Analysis and Synthesis by Linear Prediction of the Speech Wave", *J. Acoust. Soc. Am.*, Vol. 50, pp. 637-655 (1971).

7. G. von Bismarck, "Sharpness as an Attribute of the Timbre of Steady Sounds", *Acoustica*, Vol. 30, pp. 159-172 (1974).

8. D.A. Luce, "Dynamic Changes of Orchestral Instruments", *J. Audio Engr. Soc.*, Vol. 23, pp. 565-568 (1975).

9. J.W. Beauchamp, "Analysis and Synthesis of Cornet Tones using Nonlinear Interharmonic Relationships", *J. Audio Engr. Soc.*, Vol. 23, pp. 778-795 (1975).

10. D. Ehresman and D. Wessel, "Perception of Timbral Analogies", *IRCAM Report*13/78, p. 15, Paris, France [1978].

11. L.R. Rabiner and R.W. Schafer, *Digital Processing of Speech Signals*, Chapter 8, "Linear Predictive Coding of Speech", Prentice-Hall, NY, pp. 396-455 (1978).

12. Charles Dodge, "In Celebration: The Composition and its Realization in Synthetic Speech", in *Composers and the Computer*, Curtis Roads, ed., William Kaufman, Inc., Los Altos, CA, pp. 46-73 (1985).

References on Frequency Modulation Synthesis

1. L. W. Couch, *Digital and Analog Communication Systems*, Macmillan Publishing, pp. 188 -203 (1983).

2. J. Chowning, "The Synthesis of Complex Audio Spectra by Means of Frequency Modulation", *J. Audio Engr. Soc.*, Vol. 21, pp. 526-534 (1973).

3. B. Hutchins, "The Frequency Modulation Spectrum of an Exponential Voltage-Controlled Oscillator",

J. Audio Engr. Soc., Vol. 23, pp. 200-206 (1975).

4. D. Morrill, "Trumpet Algorithms for Computer Composition", *Computer Music J.*, Vol. 1, No. 1, pp. 46-52 (1977).

5. S. Saunders, "Improved FM Audio Synthesis Methods for Real-Time Digital Music Generation", *Computer Music J.*, Vol. 1, No. 1, pp. 53-55 (1977).

6. B. Truax, "Organizational Techniques for C:M Ratios in Frequency Modulation", *Computer Music J.*, Vol. 1, No. 4, pp. 39-45 (1977).

7. B. Schottstaedt, "The Simulation of Natural Instrument Tones using Frequency Modulation with a Complex Modulating Wave", *Computer Music J.*, Vol. 1, No. 4, pp. 46-50 (1977).

8. M. Le Brun, "A Derivation of the Spectrum of FM with a Complex Modulating Wave", *Computer Music J.*, Vol. 1, No. 4, pp. 51-52 (1977).

9. J. Justice, "Analytic Signal Processing in Music Composition", *IEEE Transactions on Acoustics, Speech, and Signal Processing*, pp. 670-684 (1979).

10. J. Dashow, "Spectra as Chords", *Computer Music J.*, Vol. 4, No. 1, pp. 43-52 (1980).

11. M. Rozenberg, "Linear Sweep Synthesis", *Computer Music J.*, Vol. 6, No. 3, pp. 65-77 (1982).

12. J. Chowning and D. Bristow, *FM Theory and Applications*, Yamaha Music Foundation, Tokyo, Japan (1986).

13. J. Bate, "The Effect of Modulator Phase on Timbres in FM Synthesis", *Computer Music J.*, Vol. 14, No. 3, pp. 38-45 (1990).

14. F. Holm, "Understanding FM Implementations: A Call for Common Standards", *Computer Music J.*, Vol. 16, No. 1 , pp. 34-42 (1992).

15. J. W. Beauchamp, "Will the Real FM Equation Please Stand Up", *Computer Music J.*, Vol. 16, No. 4, pp. 6-7 (1992).

16. A. Horner, J. Beauchamp, and L. Haken, "Genetic Algorithms and Their Application to FM Matching Synthesis", *Computer Music J.*, Vol. 17, No. 4, pp. 17-29 (1993).

References for Nonlinear (Waveshaping) Synthesis

1. R. A. Schaefer, "Electronic Musical Tone Production by Nonlinear Waveshaping", *J. Audio Eng. Soc.*, Vol. 18, pp. 413-417 (Aug. 1970).

2. C. Y. Suen, "Derivation of Harmonic Equations in Nonlinear Circuits", *J. Audio Eng. Soc.*, Vol. 18, pp. 675-676 (Dec. 1970).

3. R. A. Schaefer, "Production of Harmonics and Distortion in p-n Junctions", *J. Audio Eng. Soc.*, Vol.

19, pp. 759-768 (Oct. 1971).

4. J. A. Ball, "The Function Generator in Music Synthesis," *Synthesis*, Vol. 1, No. 2, pp. 29-35 (1971).

5. G. von Bismarck, "Sharpness as an Attribute of the Timbre of Steady Sounds", *Acoustica*, Vol. 30, pp. 159-172 (1974).

6. J. W. Beauchamp, "Analysis and Synthesis of Cornet Tones Using Nonlinear Interharmonic Relationships", *J. Audio Eng. Soc.*, Vol. 23, pp. 778-795 (1975).

7. C. Roads, "A Tutorial on Non-linear Distortion or Waveshaping Synthesis", *Computer Music J.*, Vol. 3, No. 2, pp. 29-34 (1979).

8. D. Arfib, "Digital Synthesis of Complex Spectra by means of Multiplication of Non-linear Distorted Sine Waves", *J. Audio Eng. Soc.*, Vol. 27, pp. 757-768 (1979).

9. M. LeBrun, "Digital Waveshaping Synthesis," *J. Audio Eng. Soc.*, Vol. 27, pp. 250-266 (1979).

10. J. W. Beauchamp, "Brass Tone Synthesis by Spectrum Evolution Matching with Nonlinear Functions", *Computer Music J.*, Vol. 3, No. 2, pp. 35-43 (1979). Republished in *Foundations of Computer Music*, C. Roads & J. Strawn, Eds., MIT Press, Cambridge, MA, pp. 95-113 (1985).

11. J. W. Beauchamp, "Practical Sound Synthesis Using a Nonlinear Processor (Waveshaper) and a High-Pass Filter", *Computer Music J.*, Vol. 3, No. 3, pp. 42-49 (1979).

12. J. W. Beauchamp, "Analysis of Simultaneous Mouthpiece and Output Waveforms of Wind Instruments", *Audio Eng. Soc. Preprint No. 1626* (1980).

13. J. W. Beauchamp, "Synthesis by Amplitude and 'Brightness' Matching of Analyzed Music Instrument Tones", *J. Audio Engr. Soc.*, Vol. 30, No. 6, pp. 396-406 (1982).

14. J. W. Beauchamp and A. Horner, "Extended Nonlinear Waveshaping Analysis/Synthesis Technique", *Proc. 1992 Int. Computer Music Conf.*, pp. 2-5 (1992).