Digital Waveshaping Synthesis*

MARC LE BRUN

Center for Computer Research in Music and Acoustics, Stanford University, Stanford, CA 94305

Waveshaping synthesis, a newly developed tool for the generation of musical sounds, is described. The method is well suited to implementation on a digital computer and is both economical and flexible. In addition it provides a unified conceptual framework for a number of nonlinear techniques, including frequency-modulation synthesis. Both the theory and practice of the method are developed fairly extensively, beginning with simple but useful forms and proceeding to more complex and richer variations.

FOREWORD

My purpose in this paper is to collect in a single document the central ideas and practical methods relating to waveshaping that I have accumulated over the past two years of exploring the topic. Because it is a compendium of various types of material, the style of presentation varies between tutorial and technical, and the orientation varies between practical and theoretical. In general I have tried to place the tutorial and practical sections prior to the technical and theoretical ones. However, in order that the ideas in the paper may be developed naturally, I have on occasion found it necessary to deviate from this ideal. To aid those readers who may wish to omit the tangential discussions, the titles of the sections containing more or less central or particularly useful material are marked with daggers.

Waveshaping has continued to be explored since the time when the basic outline of this paper became fixed. There are ongoing developments in both theory and practice, including some very interesting applications in the simulation of string and voice-like timbres. Unfortunately this research is too recent to be included here, and will have to be presented at a later time. Hopefully this exposition will be of some use to those who may wish to conduct further investigations of waveshaping.

* Manuscript received 1978 Nov. 5.

0. INTRODUCTION†

This paper describes *wave-shaping synthesis,* a newly developed tool for the generation of musical sounds. The method is well suited to implementation on a digital computer and is both economical and flexible. In addition it provides a unified conceptual framework for a number of *nonlinear* techniques, including frequency-modulation synthesis.

Central to the use of the computer as a musical instrument is the generation of signals having well-defined acoustical properties. An important factor in the utility of any synthesis technique is the degree of control that the user can exercise over the nature of the sounds it produced. Waveshaping synthesis provides the computer musician with a high degree of control over the spectrum of the output signal. It is possible with waveshaping synthesis to *exactly* match *any* particular steady-state harmonic spectrum, including those produced by natural sources such as musical instruments.

The method is easily extended so that time-varying spectra may be produced. The dynamic behavior seems to be musically valid. In one experiment a trombone note was digitized, and its spectrum was computed at the point of maximum amplitude. This information was then supplied to a program that employed waveshaping to create a synthetic tone whose spectrum varied dynamically in direct relation to the envelope of the original tone. The result sounded remarkably like the original note, even to experienced listeners. This was perhaps the first time that analysis data were used directly in the synthesis of a sound by means of a nonlinear technique.

While the success of the above-mentioned fully automatic process is encouraging, waveshaping is also a responsive tool for the creation of specific spectral profiles and dynamics. It is possible to interactively perform "fine tuning" on various timbral features with a degree of precision and perspicuity not easily obtainable with most other nonlinear methods. Work is now proceeding on employing waveshaping in the imitation of other natural instrument sounds as well as in the creation of new sounds.

Another important consideration regarding any synthesis method is its computational cost effectiveness. In this respect waveshaping, like most nonlinear methods, is quite cheap. In general the cost is close to that of frequency-modulation synthesis: a few multiplies and a handful of table lookups accomplish a great deal. In fact, as will be shown later, frequency modulation can be treated as a *special case* of waveshaping.

The last point indicates an interesting sidelight this research has thrown on the fundamental structure of many nonlinear methods—it appears that there is a unified formal basis underlying a number of the seemingly disparate nonlinear techniques. It is an exciting possibility that this theoretical unity is perhaps in some way related to the physical principles that control the production of natural acoustical phenomena. If so, it means that the great success of frequency-modulation synthesis (for example) in achieving lifelike timbres is not accidental, and that new nonlinear acoustical models may be constructed. For this reason the mathematical aspects of waveshaping will be developed fairly extensively below.

To summarize then, waveshaping combines a degree of the control and flexibility inherent in classical additive synthesis with the cost effectiveness of frequency-modulation synthesis. In addition the mathematical structure is suggestive and constitutes a theoretical tool for the analysis of nonlinear methods. It is hoped that this presentation may stimulate further research in these areas.

1. WAVESHAPING IN PRACTICE†

We will begin by describing how waveshaping is accomplished in practice and then proceed to its theoretical analysis. The intention is that this pragmatic approach will make the presentation clearer and more meaningful. The basic components of the simplest form of the waveshaping algorithm are depicted in Fig. 1.

We start with a pure sinusoidal signal that we will denote by x. To make this clearer we can define $x = \cos\theta$, with $\theta = \omega t$, where t is time and ω is the radian frequency. We choose to use the cosine in defining x since it makes the mathematics somewhat simpler. In any event x varies with time within the range $[-1+1]$, that is, at any given time $-1 \leq x \leq +1$.

The next step is the crucial one and the one that gives waveshaping its name. We take the signal x and compute some function of it that we denote by $f(x)$. The particular function f we pick will be left unspecified for now. The only constraint we will make is that f take a number in the range $[-1+1]$ and return a result in the same range (sometimes called the *signed unit interval*).

In practice this can be accomplished by means of a table lookup. We create a one-dimensional array that contains the values of f at equally spaced points through the signed unit interval and store those values in a table. When the time comes to compute $f(x)$, we look up the value corresponding to x in the f table, possibly interpolating to get the "in between" values. It can be seen that *any* reasonable function of x can be computed in this manner. An example subroutine for doing this computation is Program 1.

We will call f the *shaping function* since its effect is to change the shape of the input wave x. To see how this works, suppose that f is the identity function $f(x) = x$. In this case the signal will pass through unchanged, as depicted in Fig. 2(a). Suppose we now make a little "bump" in f near 0. Then, when the signal x becomes small in magnitude, the lookup process will output values a little larger than those input to it, and the resulting signal will acquire some sort of bump as well. This is shown in Fig. 2(b). The remainder of Fig. 2 shows the effects that some other choices for f have on the sinusoidal input signal.

It can be seen that a great variety of waves can be produced with this method. Every possible symmetric waveshape can be created in this way. If for some reason we wished to produce a particular waveform $g(\theta)$, we simply set $f(x) = g(\cos^{-1}(x))$, where $\cos^{-1}$ is of course the arccos or inverse cosine function.

2. DYNAMIC SPECTRUM GENERATION†

Up to this point we have not really accomplished much. If all we desired was a particular waveshape, we could have produced it more easily by simply reading it out of a table in the same way that sine waves and other waves are often generated. We can, however, cause the spectrum to vary in an interesting and possibly musically useful way by means of *multiplying* the input signal x by some value a in the interval $[0\ 1]$ (the *unit interval*) before applying f to it. We will then be computing the function $f(ax)$. If the value of a is varied over time, different subranges of the function f will be scanned, which will cause the output wave to have a varying shape as well. An example of a

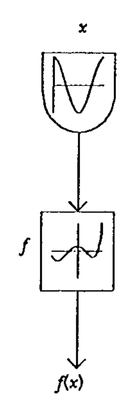


Fig. 1. Basic waveshaping synthesis system.

function shaping a sinusoid scaled by various values of a is shown in Fig. 3.

Since the waves will be different in shape, they will have differing harmonic content. The exact spectrum produced depends on the shaping function and the value of a. We will examine the formulas that specify the spectrum in terms of f and a later. For now we will assume that this is an interesting way to create signals with varying harmonic contents.

Following the usual usage in nonlinear synthesis, we will call the parameter a the *index*. If we vary the index over the duration of a note, we thereby vary the spectrum over the course of the note. In order to implement this we must specify a time-varying *index function* $a = a(t)$. This is diagrammed in Fig. 4. Since our results will make little use of the detailed dynamic behavior of the index, we will often write a with the understanding that we may vary it in order to obtain a changing spectrum.

3. SYMMETRIC, INHARMONIC, AND FOLDED SPECTRA†

There is one further refinement that we may wish to apply to our synthesis technique. As we are currently defining it, the method is only capable of generating *harmonic* spectra, that is, signals that contain only frequencies which are integer multiples of the fundamental frequency ω. We may escape this constraint somewhat by the stratagem of multiplying the output signal by a sinusoid having some arbitrary frequency C. A diagram of the technique with this operation included may be found in Fig. 5. (Note that in the special case where $C = \omega$ we do not need an additional oscillator unit since we can obtain the same effect by multiplying the output of the waveshaper by x.)

This multiplication causes every harmonic spectral line of frequency $k\omega$ and amplitude h_k in the original signal to be replaced by two spectral lines (called *sidebands*) of frequencies $C + k\omega$ and $C - k\omega$, each with amplitude $h_k/2$. We may express this mathematically as

$$\cos Ct \sum_{k=0}^{\infty} h_k \cos k\omega t = \sum_{k=0}^{\infty} h_k \cos Ct \cos k\omega t$$

$$= \sum_{k=0}^{\infty} \frac{h_k}{2} (\cos (C + k\omega)t + \cos (C - k\omega) t).$$

This spectrum is *symmetrical* about the *center* or *carrier* frequency C. (Symmetrical spectra are also sometimes called *two-sided spectra*.) Various types of choices of C produce various types of effects on the signal. If C is a rational multiple of ω, the resulting spectrum will be harmonic. If C is an irrational multiple of ω, the spectrum will contain frequencies which cannot simultaneously be overtones of any fundamental, and the output will therefore be inharmonic.

Inharmonic sounds of this type can be used to simulate the inharmonicity of many natural percussive sounds such as bells or drums. This effect has been used quite successfully in frequency-modulation FM synthesis [1].

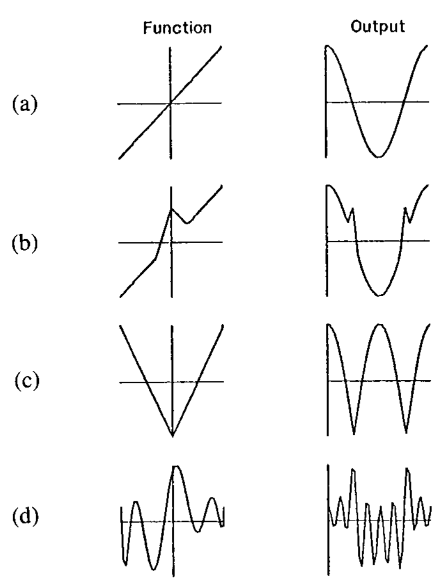


Fig. 2. Effect of various shaping functions.

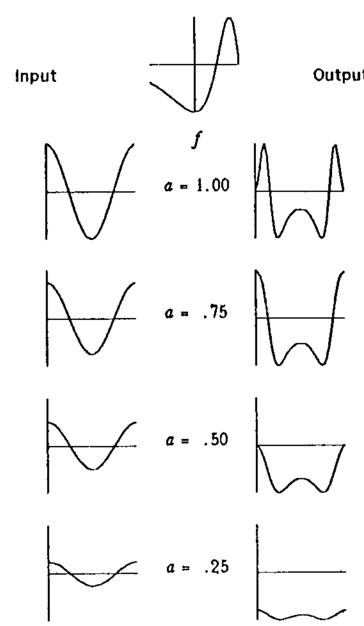


Fig. 3. Effect of a shaping function as the index varies.

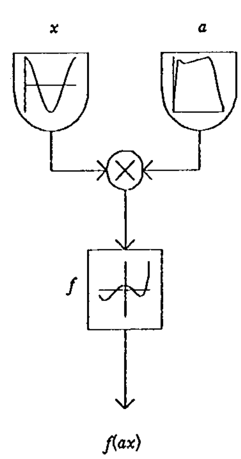


Fig. 4. Dynamic spectrum waveshaping synthesis system.

Another interesting effect occurs when the original signal contains harmonics $\cos k\omega t$ where $k > C/\omega$, that is, when there are harmonics whose number exceeds the ratio of the carrier frequency C to the frequency of the "fundamental" ω. When this happens, the frequency of the difference sideband is negative. We may ignore this negation because of the elementary trigonometric identity $\cos -\theta = \cos \theta$. These sidebands therefore "reflect" around zero, which might be called "foldunder" by analogy with the familiar foldover phenomenon. Foldunder can be used to increase the richness and dynamic complexity of both harmonic and inharmonic tones. Representations of some of these carrier effects are presented in Fig. 6.

It is interesting to note that *exactly* the same classes of spectral structures are generated by the same types of choices of carrier frequency in frequency-modulation synthesis. We will examine the reason for this later. Also, in a form of nonlinear synthesis known as *discrete summation* synthesis, described in [2], which produces symmetric spectra, we find factors corresponding to our multiplication by $\cos C$. (We shall also explore this technique later.)

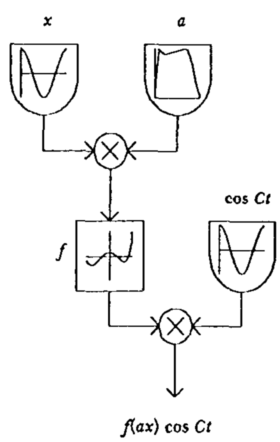


Fig. 5. Waveshaping synthesis system with carrier.

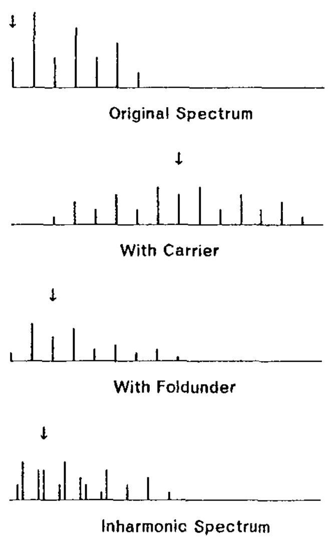


Fig. 6. Carrier effects. Arrows point to position of carrier frequency.

At this point the essential components of a waveshaping synthesis system have been outlined, and we may turn to a more detailed examination of the underlying theory which both allows us to predict the sounds produced by different shaping functions and furthermore provides us with a method for constructing shaping functions to our specifications.

4. CONSTRUCTION OF SHAPING FUNCTIONS†

We begin by deriving a series of functions T_n, which, when used as shaping functions, produce as output the pure nth harmonic. Recalling that $x = \cos \theta$, we may write this as

$$T_n(x) = T_n(\cos \theta) = \cos n\theta .$$

With this definition we can easily find expressions for T_0 and T_1:

$$T_0(x) = \cos 0\,\theta = \cos 0 = 1$$
$$T_1(x) = \cos 1\,\theta = \cos \theta = x .$$

We will now derive a *recursive* definition for T_{n+1} in terms of T_n and T_{n-1}. Since we have expressions for T_1 and T_0, this will allow us to determine T_2. We can then use T_2 along with T_1 to get T_3 and continue in the same way to construct T_4, T_5, $\cdots$, and so on.

The derivation of our recursion formula starts with the elementary trigonometric expansion of a product of cosines into a sum of sidebands:

$$\cos u \cos v = \frac{\cos(u+v) + \cos(u-v)}{2} .$$

If we now set $u = n\theta$ and $v = \theta$, this gives

$$\cos n\theta \cos \theta = \frac{\cos(n\theta+\theta) + \cos(n\theta-\theta)}{2}$$
$$= \frac{\cos(n+1)\theta + \cos(n-1)\theta}{2} .$$

If we next make our usual substitution $\cos \theta = x$ and employ our definition $\cos n\theta = T_n(\cos \theta) = T_n(x)$, we get

$$x\,T_n(x) = \frac{T_{n+1}(x) + T_{n-1}(x)}{2} .$$

Solving for T_{n+1} produces

$$T_{n+1}(x) = 2xT_n(x) - T_{n-1}(x)$$

which is the required recursive formula.

We may then employ this formula to calculate successive T_n:

$$T_0(x) = 1$$
$$T_1(x) = x$$
$$T_2(x) = 2x^2 - 1$$
$$T_3(x) = 4x^3 - 3x$$
$$T_4(x) = 8x^4 - 8x^2 + 1$$
$$T_5(x) = 16x^5 - 20x^3 + 5x$$
$$T_6(x) = 32x^6 - 48x^4 + 18x^2 - 1$$
$$T_7(x) = 64x^7 - 112x^5 + 56x^3 - 7x .$$

These functions, called the *Chebychev polynomials* (of the first kind), are well known. Many useful formulas relating to them can be found in standard references such as [3] or [4]. We will introduce various aspects which are of musical significance at the appropriate points in the subsequent discussion. Graphs of T_n for $n = 0, \ldots, 7$ appear in Fig. 7. A routine to compute the value of $T_n(x)$ is given in Program 2.

Since these functions produce only a single harmonic, we may create any steady-state spectrum we wish by taking a suitable linear combination of them. This is done in a manner similar to additive synthesis, by making a weighted sum. If we desired to create a steady-state spectrum consisting of h_1 parts of the fundamental, h_2 parts of the second harmonic, h_3 parts of the third harmonic, and so on (including $h_0/2$ parts of 0th harmonic, or constant offset, if we wish), we construct the shaping function defined by

$$f(x) = \frac{h_0}{2} + \sum_{k=1}^{\infty} h_k T_k(x).$$

(The constant term is divided by 2 for reasons we cannot go into here.) To make this clearer, let us suppose that we wish to create the shaping function f_1 which has the steady-state spectrum where $h_1 = 9, h_2 = 3, h_3 = 5, h_4 = 7$, and $h_5 = 1$ parts, respectively. Then we define f_1 as the sum

$$\begin{array}{lcrrrrrr}
9T_1(x) & = & & & & & 9x & \\
3T_2(x) & = & & & & 6x^2 & & -3 \\
5T_3(x) & = & & & 20x^3 & & -15x & \\
7T_4(x) & = & & 56x^4 & & -56x^2 & & +7 \\
+T_5(x) & = & 16x^5 & & -20x^3 & & +\ 5x & \\
\hline
f_1(x) & = & 16x^5 & +56x^4 & & -50x^2 & -x & +4
\end{array}$$

A graph of this function (scaled by its maximum value 25 = 9 + 3 + 5 + 7 + 1) appears in Fig. 8. In practice we would keep the harmonic amplitudes in a table and then compute the values for the f table automatically. An example routine which employs the recursion formula is given in Program 3.

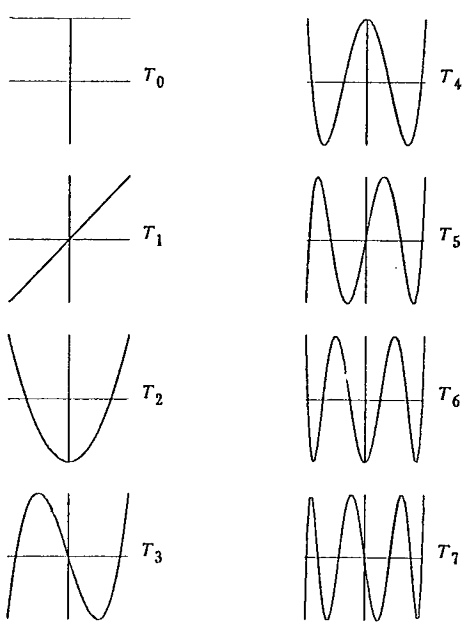


Fig. 7. Chebychev polynomials T_0 through T_7.

5. DYNAMIC SPECTRA OF SHAPING FUNCTIONS

Now that we know how to construct shaping functions which produce a particular steady-state spectrum, we would like to know about the behavior of the spectrum when we change the index a (the steady-state spectrum is of course that produced when $a = 1$). Since the harmonic amplitudes vary when a varies, we will begin to notate them as $h_k(a)$. Where we previously wrote h_k we will now write $h_k(1)$ to make explicit the dependence of the harmonic amplitudes on the index.

To state our current problem precisely, we wish, given a shaping function f defined by

$$f(x) = \frac{h_0(1)}{2} + \sum_{k=1}^{\infty} h_k(1) T_k(x)$$

to express $f(ax)$ in terms of $T_n(x)$ and $h_k(a)$, that is,

$$f(ax) = \frac{h_0(a)}{2} + \sum_{k=1}^{\infty} h_k(a) T_k(x).$$

Let us determine the dynamic spectra for an example shaping function f_2 by hand in order to get a feel for the process. We will take for our f_2 the simple function that has a steady-state spectrum which contains equal parts of the first and second harmonics, that is, where $h_1(1) = 1$ and $h_2(1) = 1$. This function is

$$f_2(x) = T_1(x) + T_2(x) = 2x^2 + x - 1.$$

Multiplying our input sinusoid x by the index a gives

$$f_2(ax) = 2a^2x^2 + ax - 1.$$

We can now essentially reverse the construction process by subtracting out successive harmonics. We will approach this a step at a time. First of all we see that $f(ax)$

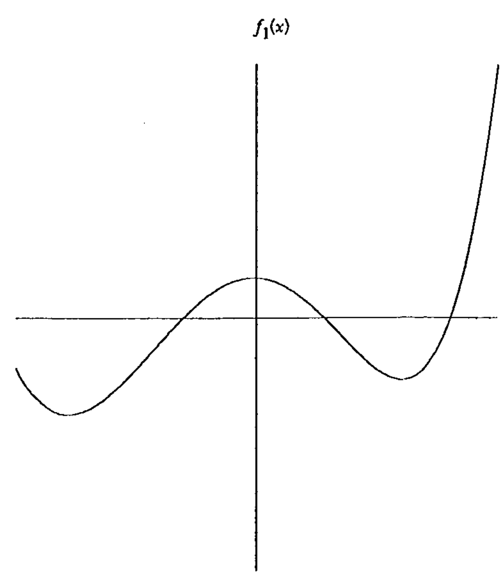


Fig. 8. Example shaping function $f_1(x)$.

cannot contain any harmonics higher than the second because the expression contains no power of x greater than the second. This suggests that we should begin by looking at the coefficient of the highest power of x. Let us express our shaping functions as power series in x:

$$f(ax) = \sum_{p=0}^{\infty} d_p(a)x^p.$$

In our example shaping function f_2 we have $d_2(a) = 2a^2$, $d_1(a) = a$, and $d_0(a) = -1$. Expressed in this way we can specify a procedure for converting a polynomial shaping function of degree M into a sum of Chebychev polynomials. The algorithm is as follows. First set $Q = f(ax)$; second, subtract from Q the expression $h_m(a)T_m(x)$, where

$$h_m(a) = \frac{q_m(a)}{2^{m-1}}$$

where m is the current maximum exponent in Q and $q_m(a)$ is the corresponding coefficient. Then, if $m > 0$, continue the process with the resulting Q (which is now of smaller degree). By "unwinding" the successive subtractions we can see that these $h_m(a)$ are in fact the correct harmonic amplitudes in our *Chebychev expansion* of $f(ax)$.

To make this process a bit more concrete, we will apply it to our example shaping function f_2. Initially we have

$$Q \leftarrow f_2(ax) = 2a^2x^2 + ax - 1.$$

Our highest power of x is $m = M = 2$, so we find

$$h_2(a) = \frac{q_2(a)}{2^{2-1}} = \frac{2a^2}{2} = a^2$$

multiplied by $T_2(x)$ and subtracted from Q leaves us with

$$Q \leftarrow Q - a^2T_2(x) = (2a^2x^2 + ax - 1) - (2a^2x^2 - a^2) = ax + (a^2 - 1).$$

Now $m = 1$ and $q_1(a) = a$, so we have

$$h_1(a) = \frac{q_1(a)}{2^{1-1}} = \frac{a}{1} = a.$$

Subtracting from Q gives

$$Q \leftarrow Q - aT_1(x) = (ax + a^2 - 1) - (ax) = a^2 - 1.$$

continuing with the final step we have $q_0(a) = a^2 - 1$, giving

$$h_0(a) = \frac{q_0(a)}{2^{0-1}} = \frac{a^2-1}{1/2} = 2(a^2 - 1).$$

We can check this expansion of f_2 by adding the terms back up in the same way as we constructed the previous shaping functions, if we so desire:

$$\frac{h_0(a)}{2} + \sum_{k=1}^{\infty} h_k(a)T_k(x)$$
$$= \frac{2(a^2-1)}{2} + aT_1(x) + a^2T_2(x)$$
$$= a^2 - 1 + ax + 2a^2x^2 - a^2$$
$$= 2a^2x^2 + ax - 1.$$

Now that we have the formulas for the h_k functions, we can predict the exact amplitude of each harmonic in our output wave for any given value of the index we choose. For instance, when $a = 0.5$ we have $h_2(0.5) = 0.25$, $h_1(0.5) = 0.5$, and $h_0(0.5)/2 = -0.75$.

Graphs of the harmonic amplitudes with respect to the index appear in Fig. 9(a). Note that since a *negative* amplitude for a given harmonic means that the phase of that harmonic is inverted, we may choose to plot the absolute value in our graphs since it facilitates comparison of such things as the perceived amplitude. The absolute values of these functions are in Fig. 9(b).

For a more elaborate example of these processes we may expand the earlier fifth-order shaping function f_1. The harmonic amplitudes in that case are given by

$$h_0(a) = 2\cdot(21a^4 - 25a^2 + 4)$$
$$h_1(a) = 10a^5 - a$$
$$h_2(a) = 28a^4 - 25a^2$$
$$h_3(a) = 5a^5$$
$$h_4(a) = 7a^4$$
$$h_5(a) = a^5.$$

Graphs of these functions appear in Fig. 10 rectified as described above for Fig. 9.

6. POWER SERIES AND CHEBYCHEV EXPANSIONS OF SHAPING FUNCTIONS†

We have now gained a little experience with deriving the harmonic amplitude functions that arise from a given shaping function. To summarize, we found it convenient to describe our shaping function in terms of a power series. We then "unwound" this power series coefficient by coefficient to get the corresponding Chebychev expansion. This suggests that it might be useful to have a way of describing the Chebychev expansion in terms of the power series, and, conversely, a way of describing the power series in terms of the Chebychev expansion.

These dual descriptions were originally found by observation and inference in [5] and later extended to the

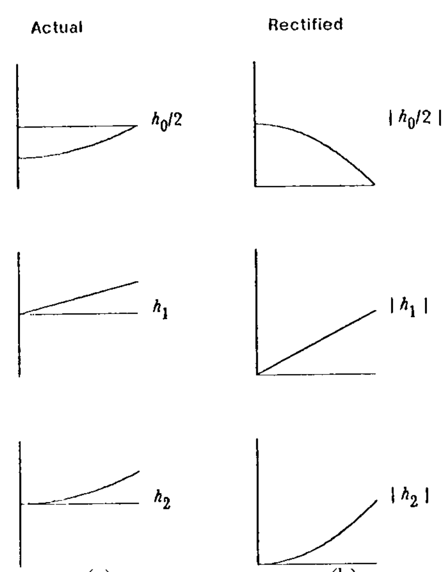


Fig. 9. Harmonic amplitudes for example shaping function f_2.

dynamic case in [6]. We may derive such formulas analytically by using the De Moivre and binomial theorems and constructing expansions for the Chebychev polynomials by means of Gauss's hypergeometric identity, using trigonometric identities to express powers of x, combining the sums, and simplifying them. We will omit these derivations, noting in passing that their analytic nature will allow us to more easily construct variations. The respective expansions are

$$h_k(a) = 2 \sum_{j=0}^{\infty} \binom{k+2j}{j} \times \frac{1}{2^{k+2j}} a^{k+2j} d_{k+2j}(1)$$

for the Chebychev and

$$d_p(a) = 2^p - 1a^p \sum_{i=0}^{\infty} -1^i \left(\binom{p+i}{i} + \binom{p+i-1}{i-1} \right) h_{p+2i}(1)$$

for the power series.

It might be noted that for the cases that arise with $i = 0$ $\binom{p-1}{-1} = 0.$

With these formulas we are able to express the dynamic behavior of the shaping function entirely in terms of its steady-state spectrum. To do this we cancel out the d coefficients in the first equation by substituting for them the entire right-hand side of the second equation with a set equal to 1 and p set to $k + 2j$. After the various powers of 2 cancel, we are left with the central equation:

$$h_k(a) = \sum_{j=0}^{\infty} \binom{k+2j}{j} a^{k+2j} \sum_{i=0}^{\infty} -1^i \left(\binom{k+2j+i}{i} + \binom{k+2j+i-1}{i-1} \right) \times h_{k+2j+2i}(1).$$

With this equation we can completely predict the dynamic behavior of the spectrum for a particular shaping function given its steady-state spectrum. To do this we construct the $h_k(a)$ functions and then compute their values for any a we wish. An examination of this central equation leads to a number of insights. Some of these insights are of practical importance and some of them are of theoretical interest.

One notational simplification is the converse substitution of p for $k + 2j$ everywhere in the central equations given above. This leads to the form

$$h_k(a) = \sum_{j=0}^{\infty} \binom{p}{j} a^p \sum_{i=0}^{\infty} -1^i \left(\binom{p+i}{i} + \binom{p+i-1}{i-1} \right) h_{p+2i}(1) \qquad p = k + 2j.$$

Note that the inner summations contain only p and the summation index explicitly. This means that we can define some auxiliary variables D_p to stand for this inner summation. This produces the simple form

$$h_k(a) = \sum_{j=0}^{\infty} \binom{p}{j} D_p a^p \qquad p = k + 2j.$$

To make all this a bit clearer we can write out a few of these sums. First the D_p:

$$D_0 = \left(\binom{0}{0} + \binom{-1}{-1}\right) h_0 - \left(\binom{1}{1} + \binom{0}{0}\right) h_2 + \left(\binom{2}{2} + \binom{1}{1}\right) h_4 - \cdots$$

$$D_1 = \left(\binom{1}{0} + \binom{0}{-1}\right) h_1 - \left(\binom{2}{1} + \binom{1}{0}\right) h_3 + \left(\binom{3}{2} + \binom{2}{1}\right) h_5 - \cdots$$

$$D_2 = \left(\binom{2}{0} + \binom{1}{-1}\right) h_2 - \left(\binom{3}{1} + \binom{2}{0}\right) h_4 + \left(\binom{4}{2} + \binom{3}{1}\right) h_6 - \cdots$$

$$D_3 = \left(\binom{3}{0} + \binom{2}{-1}\right) h_3 - \left(\binom{4}{1} + \binom{3}{0}\right) h_5 + \left(\binom{5}{2} + \binom{4}{1}\right) h_7 - \cdots$$

The h functions are then

$$h_0(a) = \binom{0}{0} D_0 a^0 + \binom{2}{1} D_2 a^2 + \binom{4}{2} D_4 a^4 + \cdots$$

$$h_1(a) = \binom{1}{0} D_1 a^1 + \binom{3}{1} D_3 a^3 + \binom{5}{2} D_5 a^5 + \cdots$$

$$h_2(a) = \binom{2}{0} D_2 a^2 + \binom{4}{1} D_4 a^4 + \binom{6}{2} D_6 a^6 + \cdots$$

$$h_3(a) = \binom{3}{0} D_3 a^3 + \binom{5}{1} D_5 a^5 + \binom{7}{2} D_7 a^7 + \cdots$$

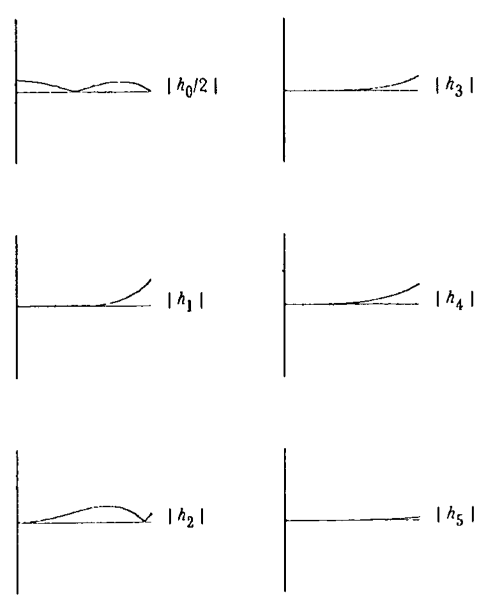


Fig. 10. Harmonic amplitudes for example shaping function f_1.

For our simple example function f_2 we have

$$D_0 = (1 + 0) \cdot 0 - (1 + 1) \cdot 1 = -2$$
$$D_1 = (1 + 0) \cdot 1 = 1$$
$$D_2 = (1 + 0) \cdot 1 = 1$$

and

$$h_0(a) = 1 \cdot -2 \cdot 1 + 2 \cdot 1 \cdot a^2 = 2(a^2 - 1)$$
$$h_1(a) = 1 \cdot 1 \cdot a = a$$
$$h_2(a) = 1 \cdot 1 \cdot a^2 = a^2$$

which agrees with our previous results.

The interested reader may wish to verify that for the more complex fifth-order example shaping function f_1 we obtain $D_0 = 8, D_1 = -1, D_2 = -25, D_3 = 0, D_4 = 7$, and $D_5 = 1$ (which lead necessarily to the h functions given before).

A number of features of this equation are apparent at this point. For instance, even though the sums are notated as infinite power series, we see that they are really finite polynomials if the steady-state spectrum is band-limited. That is, if in the original shaping function all $h_k(1)$ with k greater than some M are zero, then the same will be true of all $h_k(a)$ with $k > M$.

Another property is that the subscripts and powers in the sums effectively increment by twos, thus the odd harmonics and the even harmonics are independent and are only influenced by harmonics of the *same parity*. Furthermore, the indices (and powers) effectively run from *k upwards* through M, which means that the dynamic behavior of a given harmonic is determined only by the steady-state values for that harmonic and those above it (of the same parity).

This last observation may be interpreted as meaning that as we decrease the value of a from unity, the "energy" that was originally present in a given harmonic is gradually transferred to lower harmonics. When the index drops all the way to zero, the only remaining harmonic is the zeroth. If a carrier is being used, the resulting signal is pure carrier. Thus, as with frequency-modulation synthesis, the "bandwidth" of the output signal varies directly with the index, although the individual harmonics vary in a somewhat complicated manner.

Taking account of these features we can produce a moderately efficient subprogram, such as that given in Program 4, for constructing tables containing the coefficients of the a polynomials and for employing those tables in evaluating the $h_k(a)$ functions.

7. WAVESHAPING INTERPRETATION OF FREQUENCY-MODULATION SYNTHESIS†

As was mentioned earlier, it is possible to treat frequency-modulation synthesis as a form of waveshaping. By casting the better known techniques of frequency modulation in the forms we are dealing with here, we may increase our understanding of waveshaping. In addition we will perhaps gain some further insight into frequency-modulation synthesis due to viewing it in a novel manner. Therefore, before continuing with our theoretical investigation of waveshaping, we will briefly examine frequency modulation in this context.

The pertinent formula describing the frequency-modulation synthesis algorithm is

$$\mathrm{FM} = \cos(Ct + I \cos \omega t)$$

with C being the center or carrier frequency, ω the modulating frequency, I the modulation index and t, of course, time. The effect of the Ct term is to cause the signal to be symmetric about some carrier. The creation of signals with symmetry about a carrier is easily accomplished by multiplying any signal by a sinusoid whose frequency is the carrier frequency we desire. Since this is a nonessential (though useful) component of frequency modulation, we will simply remove it from the expression, leaving

$$\cos(I \cos \omega t).$$

Next we will perform our (by now) familiar change of variable $x = \cos \omega t$, giving

$$\cos(Ix).$$

Let us denote the maximum value of the modulation index by I_{max}. We may then replace I with $I_{max}\, a$, where a will lie in the unit interval. This gives

$$\cos(I_{max} ax).$$

Finally we can replace the cosine with a function f_{FM} defined by

$$f_{FM}(x) = \cos(I_{max} x)$$

leaving us with

$$f_{FM}(ax)$$

which is clearly in the format of our waveshaping expression. If we desire a carrier, we can reintroduce it at this point by computing the product

$$\cos(Ct) f_{FM}(ax).$$

In summary then, frequency modulation may be treated as a form of waveshaping where the function in the shaping table is I_{max}/π cycles of a cosine. It should be noted that we have implemented an *imitation* of frequency modulation here, ignoring, for example, phase effects. It is not too much more difficult to implement frequency modulation *exactly*. The details would have unduly complicated this discussion, however. We thus discover that frequency-modulation synthesis may be treated, if we wish, as a special case of waveshaping synthesis.

As a concrete exercise we might take the example of a brass-like tone given in [1]. Here $I_{max} = 5$, so we fill our table with the values of shaping function $f(x) = \cos(5x/\pi)$ for x in the range $[-1\ +1]$ and scale the index function into the unit interval. Since a 1/1 frequency ratio carrier is used, we will also multiply the output of the shaper by x. Other particular frequency-modulation instruments may be implemented in a similar manner.

8. WAVESHAPING INTERPRETATION OF DISCRETE SUMMATION SYNTHESIS

Another nonlinear technique is one presented in [2]. As with frequency modulation, it may be implemented as a waveshaping operation. Discrete summation is more recent than frequency modulation and is less well known as a means of producing musical sounds. While it therefore has less practical import, it is of particular theoretical interest because it is possible to derive closed-form expressions for a number of its significant features. This is very helpful in developing analytical techniques and often evokes insights into the underlying structure of the synthesis systems involved.

Of the several discrete summations given in [2] we will examine one which produces a "band-unlimited" spectrum since it is of a mathematically simpler form. The relevant summation describing the output wave is

$$f(\cos \omega t) = \sum_{k=0}^{\infty} r^k \cos k\omega t.$$

In terms of our notation this specifies a wave whose steady-state spectrum is given by successive powers of r:

$$h_0(1) = 2, \qquad h_k(1) = r^k, \qquad k > 0$$

transforming the defining sum by means of our standard change of variable, gives

$$f(x) = \sum_{k=0}^{\infty} r^k T_k(x).$$

We may think of this function as the generating function for the Chebychev polynomials in x in terms of the parameter r. This sum has a closed form which is easily derived using the recurrence formula for the Chebychev polynomials:

$$f(x) = \frac{1 - rx}{1 + r^2 - 2rx}.$$

Shaping the input sinusoid x with this function will result in a steady-state signal having the exponential harmonic structure given above. If we were to perform the standard change of variable on this expression in the reverse direction, we would get the closed form stated in [2]:

$$f(\cos \omega t) = \frac{1 - r \cos \omega t}{1 + r^2 - 2r \cos \omega t}.$$

As can be seen from this example, one advantage of the waveshaping framework is that the basic structure of such expressions becomes much more apparent when it is freed of all the unsightly trigonometry.

The synthesis technique outlined in [2] achieves spectral dynamics by varying the value of r. In waveshaping synthesis we get such spectral dynamics by varying the index a in the function

$$f(ax) = \frac{1 - rax}{1 + r^2 - 2rax}.$$

Because the r^2 term in the denominator is free of a, these two methods are clearly *not* the same. To determine the exact nature of the harmonic amplitude functions for the waveshaping form, we analytically evaluate the general sums given previously for these functions. To simplify the manipulations we introduce the following rearrangement:

$$\begin{aligned} D_p &= \sum_{i=0}^{\infty} -1^i \left(\binom{p+i}{i} + \binom{p+i-1}{i-1} \right) h_{p+2i}(1) \\ &= \sum_{i=0}^{\infty} -1^i \binom{p+i}{i} (h_{p+2i}(1) - h_{p+2i+2}). \end{aligned}$$

This allows us to shift the term complexity from the binomials onto the h_k's. In the case of the discrete summation spectrum

$$h_p(1) = r^p, \qquad p > 0$$

so that

$$h_p(1) - h_{p+2}(1) = r^p - r^{p+2} = (1 - r^2) r^p, \qquad p > 0.$$

(We will be omitting the zeroth harmonic from our derivations because it involves slightly more complex expressions. We will give only the final expression for its dynamics at the end.) We now wish to evaluate the inner sum

$$\begin{aligned} D_p &= \sum_{i=0}^{\infty} -1^i \binom{p+i}{i} (1 - r^2) r^{p+2i} \\ &= (1 - r^2) r^p \sum_{i=0}^{\infty} -1^i \binom{p+i}{i} r^{2i}, \qquad p > 0 \end{aligned}$$

which can be seen to be an expansion of a reciprocal power:

$$D_p = (1 - r^2) r^p (1 + r^2)^{-(p+1)}, \qquad p > 0.$$

This can be rearranged into

$$D_p = \left(\frac{1 - r^2}{1 + r^2} \right) \left(\frac{r}{1 + r^2} \right)^p, \qquad p > 0.$$

The outer summation giving our dynamic harmonic amplitudes is therefore

$$h_k(a) = \sum_{j=0}^{\infty} \binom{k + 1j}{j} \left(\frac{1 - r^2}{1 + r^2} \right) \times \left(\frac{r}{1 + r^2} \right)^{k+2j} a^{k+2j}, \qquad k > 0.$$

This sum is more difficult. It can be expressed, however, in terms of a hypergeometric function in the following way:

$$h_k(a) = \left(\frac{1 - r^2}{1 + r^2} \right) \left(\frac{ar}{1 + r^2} \right)^k \times {}_2F_1 \left(\frac{k+1}{2}, \frac{k+2}{2}; k + 1 \left(\frac{2ar}{1 + r^2} \right)^2 \right), \quad k > 0.$$

Luckily hypergeometric functions of this form have a known equivalent in terms of elementary functions. After a bit of rearranging of the resulting expressions we get

$$h_0(a) = \frac{1 - r^2}{R} + 1, \qquad h_k(a) = \frac{1 - r^2}{R} \times \left(\frac{2ar}{1 + r^2 + R}\right)^k, \qquad k > 0$$

where the value of the auxiliary variable R is given by

$$R = \sqrt{(1 + r^2)^2 - (2ar)^2} .$$

Although this derivation was a bit cumbersome, examination of the result reveals an interesting property: for any fixed a the harmonic amplitudes are in a geometric progression. This is the essential property of the discrete summation synthesis technique, and it allows us to construct an implementation using waveshaping.

To do this we initially pick some r less than unity, but at least as large as the maximum value it takes on in a given discrete summation instrument. This value of r is then fixed and used to construct the indicated shaping table. Then for each *dynamic* values of r we pick the value of a that causes the right-hand factor in the harmonic amplitude formula to be equal to this value of r. (While it may be possible to determine analytically the value of a as a function of r, it is probably simpler to search a table of values in some appropriate manner.)

This process will exactly match the output of the corresponding discrete summation system (except for the overall scaling introduced by the left-hand factor). We can eliminate this scaling with a *normalization function* which we will discuss shortly. In summary, then, we have (by means of a theoretical analysis) brought a form of discrete summation synthesis into the framework of waveshaping. The more complex forms may also be so interpreted, but we shall not do so here.

9. NORMALIZATION†

One effect the reader may have observed (for instance, in Fig. 3) is that when a is small, the amplitude of the output signal is often also small. In fact, the amplitude usually varies with the index for most shaping functions. To ameliorate the problem of undesirable variation of amplitude which accompanies the desirable variation in timbre, it is necessary to dynamically *normalize* the output signal. To compensate for the amplitude variation we employ a *normalization function* $N_f(a)$ which, as the notation suggests, is a function of a, with the specific form depending on f. We divide the output of the wave shaper by this normalization, generating the signal:

$$\frac{f(ax)}{N_f(a)} .$$

The values of the normalization function would generally be stored in a table and a lookup operation done for the current value of a. The extended synthesis system is diagrammed in Fig. 11 and coded in Program 5. There is of course the practical question of the occurrence of a division by zero. To handle this simply return the numerator signal. If no interpolation is done in the normalization lookup, it is also possible to replace zero values in the normalization table by a value of 1, but this is less elegant.

Because the normalization function may be thought of as compensating for the amplitude variation, the somewhat more descriptive term *compensation function* is suggested in [7] (where it is reported that good results may also be obtained *without* such compensation). The term *normalization* is employed here following the usage in [2] and elsewhere. Unfortunately the word becomes somewhat overworked and possibly confused with other kinds of normalization. Nevertheless we will follow tradition here.

We can choose among several types of normalization. Probably the most musically valid is *loudness normalization,* in which the perceived volume is made to be constant for all values of the index. Since this involves psychoacoustic interactions, this type of normalization is highly complex, context dependent, irregular, and a bit too difficult for truly pragmatic implementation. Another choice, which has been utilized in discrete summation synthesis, is *power normalization*. In this case we employ the normalization function defined by the root mean square of the harmonic amplitudes, that is

$$N_f(a) = \sqrt{\frac{h_0^2(a)}{4} + \sum_{k=1}^{\infty} h_k^2(a)} .$$

In an actual situation where we have previously extracted the $h_k(a)$ functions this calculation is not too difficult. Also, in analysis of a specific shaping function it tends to lead to a closed-form expression. For example, we may express the power normalization function for the fifth-order example shaping function f_1 given earlier as

$$\sqrt{126a^{10} + 1274a^8 - 2470a^6 + 1418a^4 - 199a^2 + 16}$$

Another interesting method is *peak normalization*. Here the absolute value of the peak output of the waveshaper for a given a is scaled to unity. This sort of normalization naturally occurs in frequency-modulation synthesis where the output signal tends to have peaks of constant height. It is easy to construct the peak normalization function given the shaping function f. For any given value of a the input signal ax varies between $-a$ and $+a$. The output signal therefore takes on the values of the shaping function over that range. In particular the *absolute peak* value of the output signal is the *absolute maximum* value of f in that range. To normalize the signal we need only to divide by these peak values. This allows us to define the peak normalization function by

$$N_f(a) = \max_{-a \le \alpha \le +a} |f(\alpha)|$$

In practice we construct the peak normalization function in a one-dimensional array in the following way. First set the element corresponding to $a = 0$ to $|f(0)|$. Assume that this is the middle element of the f array. The next element of the N array is a three-way maximum of the absolute values of the pair of elements adjacent to $f(0)$ in the f array and the first element of the N array. The rest of the array is constructed in a similar manner: taking the maximum of the absolute values of pairs of entries in the f array (symmetric about the $a = 0$ element) and the previous element in the N array. A version of this algorithm for constructing the N array is given in Program 6. Peak normalization functions are always nondecreasing functions of a. They often look rather peculiar, consisting of short segments of f connected by horizontal plateaus. The peak normalization functions corresponding to some shaping functions are shown in Fig. 12.

We can also determine the peak normalization function for the waveshaping implementation of frequency modulation. If the shaping function is $\cos Ix$ then, since $f(0) = 1$, the entire peak normalization is identically 1. Now consider a variation where the shaping function is $\sin Ix$. Here the peak normalization function begins at 0 but rises rapidly to 1. These facts lead to an interesting observation about frequency modulation: either the output signal is always peak normalized or else it fails to be peak normalized only for small values of the modulation index ($I \leq \pi/2$).

Since we can generate musically valid frequency-modulation signals without any need for peak normalization, the question arises as to whether it is possible to do the same in some cases for waveshaping. When we can dispense with the normalization step, computational cost will be reduced. It turns out that we can, in a somewhat ad hoc manner, cause a given mix of harmonic amplitudes to have a more nearly flat peak normalization function.

To accomplish such a transformation requires some free parameters. One such set is the *signs* of the harmonic amplitudes. We have been assuming that all the $h_k(1)$ values are positive since the sign of a harmonic usually has little auditory importance (although foldunder may cause significant effects). By manipulating the signs of the harmonics we can alter the morphology of the shaping function (and thereby its peak normalization function) without changing its harmonic content.

If we examine the lowest order (constant) term in each of the Chebychev polynomials we find that, for increasing k, their signs follow the pattern $+, \emptyset, -, \emptyset, +, \emptyset, -, \emptyset, \cdots$. This means that if we set the signs of the *even* harmonic amplitudes to the pattern $+, -, +, -, \cdots$, the resulting pattern will be $+, \emptyset, +, \emptyset, \cdots$. At zero the shaping function has a value equal to the sum of all the even harmonic amplitudes. Setting the signs so that these terms are all positive maximizes this value. The peak normalization function for an index of zero also has the same value, so we have maximized it as well. Since the peak normalization function is always nondecreasing but bounded above by unity, we have narrowed its range and thereby flattened it somewhat.

We can go further in this vein by examining the signs of the second to lowest order (linear) terms in the Chebychev polynomials. The pattern here is $\emptyset, +, \emptyset, -, \emptyset, +, \emptyset, -, \cdots$. By setting the signs of the *odd* harmonic amplitudes in an alternating pattern, they also all become positive. Here we have maximized the *derivative* of the function at zero. Hopefully this will cause the function to increase rapidly at the beginning and further narrow its possible range over the rest of the interval. This is the sort of pattern we get for the frequency-modulation case mentioned earlier.

The combined pattern of signs we wish to use is therefore $+, +, -, -, +, +, -, -, \cdots$. While this is clearly a heuristically motivated method and is not guaranteed optimal, it often does have the beneficial effect of flattening the normalization function. We call this operation on the signs of the harmonic amplitudes *signification,* a term chosen for its confusing pronunciation. Graphs of the signified forms of the shaping functions of Fig. 12 together with their peak normalization functions are given in Fig. 13. It can be seen that there is often an improvement in the flatness of the peak normalization function.

In analysis of a known shaping function f we can sometimes construct a closed-form expression for the signified form of f, which we write as $f^{\pm}$. For instance, we may signify the discrete summation formula used earlier by remembering that it is the generating function for the Chebychev polynomials in x in terms of the parameter r.

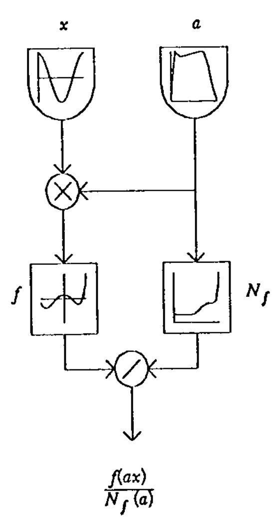


Fig. 11. Waveshaping synthesis system with peak amplitude normalization.

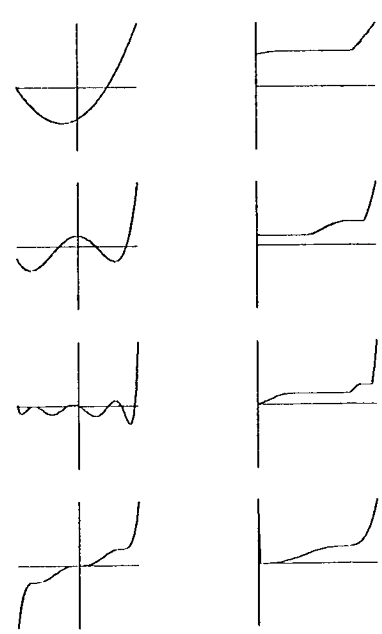


Fig. 12. Some shaping functions and their peak normalization functions.

We can then employ the power series transformation

$$g(r) \rightarrow \frac{1-\mathrm{i}}{2} g(\mathrm{i}r) + \frac{1+\mathrm{i}}{2} g(-\mathrm{i}r)$$

where i is the imaginary unit. Note that even though this transformation involves complex values, it produces a purely real result. After applying this transformation and a bit of rearrangement, we get the signified version of discrete summation:

$$f^{\pm}(x) = \frac{(1 - r^2) + rx(1 + 2rx + r^2)}{(1 - r^2)^2 + (2rx)^2} .$$

Graphs of the ordinary and signified forms of the discrete summation function, together with their associated peak normalization functions, appear in Fig. 14. It should be pointed out that while the steady-state spectrum produced by a signified function is identical to that of the original function, the dynamic behavior may be quite different. The necessary alterations to the various formulas are fairly easy to derive, so we will not do so here. Furthermore, there is a natural extension of signification by which the waveshaping synthesis system may be generalized in an interesting way.

10. PHASE QUADRATURE

The method of signification developed in the last section was based on the operation of inverting the phase of some of the harmonics. We can go further in this direction and generate signals whose harmonics have not only any steady-state amplitudes H_k we like, but which also have any arbitrary phase ϕ_k as well. This is accomplished by means of *phase quadrature*. The central expression is the elementary trigonometric identity

$$\cos u \cos v + \sin u \sin v = \cos(u - v).$$

We then take $u = k\theta$ and $v = \phi_k$, thus forming

$$\cos k\theta \cos \phi_k + \sin k\theta \sin \phi_k = \cos(k\theta - \phi_k).$$

As usual we will write T_k for $\cos k\theta$ and introduce a new function S_k for $\sin k\theta$. We then have

$$T_k \cos \phi_k + S_k \sin \phi_k = \cos(k\theta - \phi_k).$$

Finally we set h_k to $H_k \cos \phi_k$ and $\bar{h}_k$ to $H_k \sin \phi_k$, giving

$$h_k T_k + \bar{h}_k S_k = H_k \cos(k\theta - \phi_k).$$

We use the overbar to indicate a quadrant rotation of some sort. We will also employ the symbol y to indicate the sine signal:

$$y = \bar{x} = \pm \sqrt{1 - x^2} = \sin \theta.$$

Returning to the previous equation, we now wish to determine the nature of the S_k functions. We can do this in exactly the same manner as was done for the T_k functions earlier. Remembering that

$$S_n = \sin n\theta$$

we find that the expressions for S_0 and S_1 are

$$S_0 = \sin 0\,\theta = \sin 0 = 0$$
$$S_1 = \sin 1\,\theta = \sin \theta = y.$$

Continuing as before, we find that the S_n functions satisfy *exactly* the same recurrence relations as did the T_n functions:

$$S_{n+1} = 2xS_n - S_{n-1}.$$

Notice in particular that the multiplier for the S_n term is x and *not* y!

The x factor generates polynomials under the recursion.

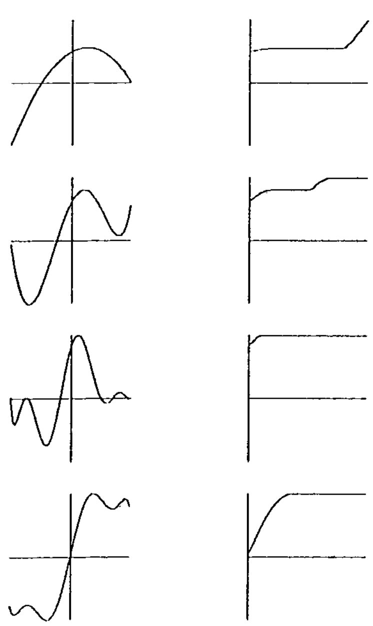


Fig. 13. Some signified shaping functions and their peak normalization functions.

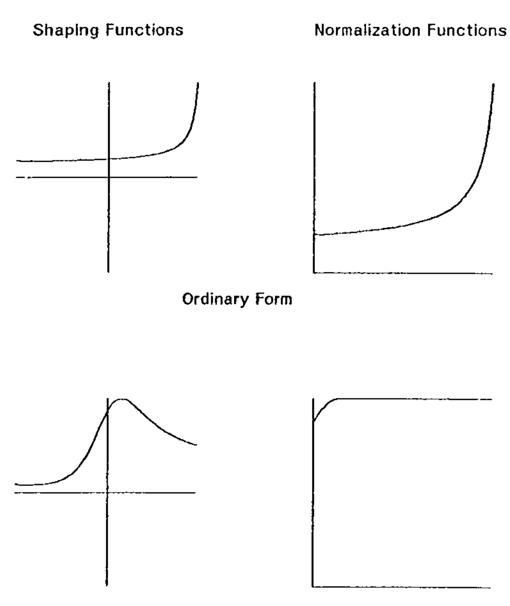


Fig. 14. Effect of signification on discrete summation and its normalizations ($r = 0.7$).

The resulting functions are

$$\begin{aligned}
S_0 &= 0\\
S_1 &= y\\
S_2 &= y \cdot (2x)\\
S_3 &= y \cdot (4x^2 - 1)\\
S_4 &= y \cdot (8x^3 - 4x)\\
S_5 &= y \cdot (16x^4 - 12x^2 + 1)\\
S_6 &= y \cdot (32x^5 - 32x^3 + 6x)\\
S_7 &= y \cdot (64x^6 - 80x^4 + 24x^2 - 1).
\end{aligned}$$

The polynomial parts of these expressions are termed the *Chebychev polynomials of the second kind* and are written U_{n-1}. Graphs of several of them appear in Fig. 15. Naturally they obey the same recurrence relation. The S_n functions are consequently defined by

$$S_n = y \cdot U_{n-1}(x).$$

To implement the phase quadrature form of waveshaping, we employ the signal y which is easily generated at the same time as x. We create another shaping function $\bar{f}$ which is specified as the weighted sum

$$\bar{f}(x) = \sum_{k=0}^{\infty} \bar{h}_k U_{k-1}(x).$$

We then multiply the output of this waveshaper by y and add it to the output of the regular cosine waveshaper to get the phase-rotated signal we desire. To get dynamic spectra we scale *both* x and y by a.

A routine to compute f and $\bar{f}$ in this form given arrays with the H and ϕ values can be found in Program 7. A phase quadrature waveshaping algorithm using these techniques is given in Program 8 and is diagrammed in Fig. 16. It allows us to produce an output signal with any set of steady-state harmonic amplitudes and phases. Note that we omit peak normalization with phase quadrature in our implementation as it involves a number of sticky mathematical details.

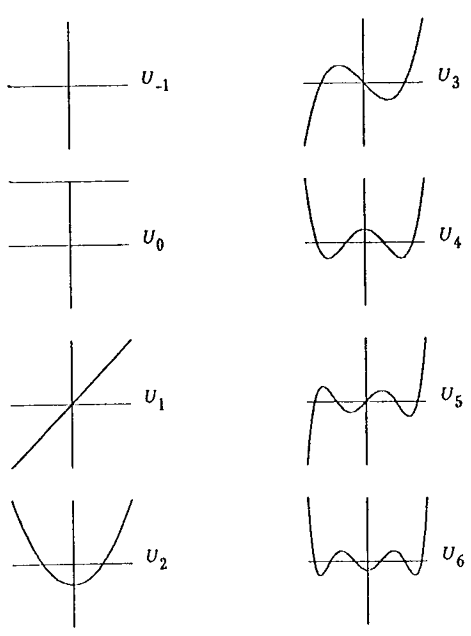


Fig. 15. Chebychev polynomials of the second kind U_{-1} through U_6. Functions are scaled by their value at 1: $U_{n-1}(1) = n$.

Like the corresponding formulas for h_k and D_p, the dynamic behavior of the sine waveshaper can be derived from De Moivre's theorem. The formulas are

$$\bar{h}_k = \sum_{j=0}^{\infty} \left(\binom{k+2j-1}{j} - \binom{k+2j-1}{j-1} \right) a^{k+2j} \bar{D}_{k+2j}$$

with

$$\bar{D}_p = \sum_{i=0}^{\infty} -1^i \binom{p+i-1}{i} \bar{h}_{p+2i}(1).$$

We have very little practical experience with the phase quadrature waveshaper and cannot offer many useful observations. One unexpected feature is that the $\bar{h}_k$ functions are "tamer," that is, have a smaller magnitude of variation than the corresponding h_k functions for a given harmonic mix. In general, as the index a approaches zero, the cosine terms seem to dominate the sine terms in their contribution to the overall dynamic behavior. This seems reasonable since all the ϕ_k approach zero.

Phase quadrature extends the waveshaping system into the complex domain, often a mathematically fruitful transformation. For instance, we can take De Moivre's theorem and construct nonrecursive closed-form expressions for the T_n and S_n functions. They are

$$T_n(x) = \frac{(x + \sqrt{x^2 - 1})^n + (x - \sqrt{x^2 - 1})^n}{2}$$

$$S_n(x) = \frac{(x + \sqrt{x^2 - 1})^n - (x - \sqrt{x^2 - 1})^n}{2\mathrm{i}}$$

where i is again the imaginary unit. (A similar form for the analogous tangent function (equal to S_n / T_n) is given in [8]. These expressions may be quite helpful in analysis where the recursive definitions may be completely unsuitable.

11. OPEN QUESTIONS

There are a number of questions open for future inquiry. Some of these are theoretical in nature. For instance, in [9] we find mention of a formula which we might write as

$$\mathrm{e}^{ax} \sin ay = \sum_{k=0}^{\infty} S_k(x) \frac{a^k}{k!}.$$

It is the factorial generating function for the S_k functions. This sum describes a variation of frequency modulation which does not involve Bessel functions in its expansion. Also, it is a generating function where the index itself is the parameter. Symbolic integration of the ordinary generating function produces the exotic-looking, but possibly useful, function

$$1 - \frac{1}{2} \ln(1 + r^2 - 2rx) = \sum_{k=0}^{\infty} T_k(x) \frac{r^k}{k}.$$

These intriguing closed forms may have utility in some nonlinear synthesis schemes. As was pointed out in [2], the number of such formulas which can be derived or which lie buried in reference works is vast. One tool for unearthing them is the Fourier series transform which has undergone our usual change of variable. This allows us to find the Chebychev series expansion of a function directly. The coefficients are given by

$$h_k = \int_{-1}^{+1} \frac{f(x)}{y} \; T_k(x)\,\mathrm{d}x \qquad \text{and}$$

$$\bar{h}_k = \int_{-1}^{+1} \frac{f(x)}{y} \; S_k(x)\,\mathrm{d}x .$$

Probably the most interesting theoretical questions involve the problem of mimicking the dynamic behavior of a given sound to produce a reasonable facsimile of that sound. One of the most significant features of waveshaping is its ability to match particular spectral mixes. That a deeper understanding of its structure may lead to the formulation of a similarly cost effective technique capable of good dynamic approximations is an exciting possibility.

More pragmatic questions concern the issues of accuracy, resolution, quantization, approximation, and interpolation in the waveshaping algorithm. Other areas now under investigation include replacing the sinusoid x by more complex signals and the generation of formants and other particular spectral features. The former method has been used with a great deal of success in frequency-modulation synthesis where (because of the general intractability of the Bessel functions) it is often the only way to achieve certain musical effects (see [10] for details). As with frequency modulation, waveshaping will produce output spectra that consist of frequencies which are sums and differences of harmonics of the input frequencies. This feature is common to a large class of nonlinear synthesis techniques. They therefore might also be called *multiplicative* synthesis techniques since such spectra always arise in some way from signal multiplication.

Recently experiments have been conducted using waveshaping in the construction of formants and other spectral structures. The straightforward nature of the system has allowed a great deal of flexibility in setting up and controlling the signals needed in such applications as the production of voicelike timbres. For example, it is possible using phase quadrature to produce formants with asymmetric contours and other arbitrary structural properties. The results of these experiments are interesting from both a theoretical and a musical viewpoint.

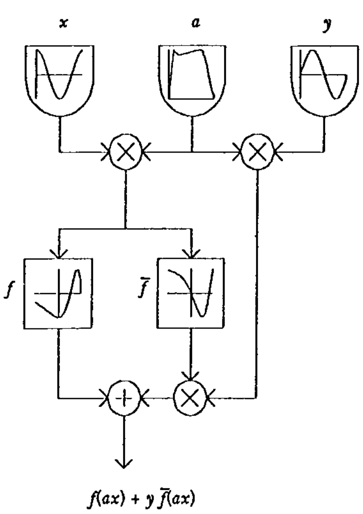


Fig. 16. Phase quadrature waveshaping synthesis system.

There are numerous other issues worth investigation. For instance, how can we predict the qualitative aspects of the shaping function given the steady-state harmonic amplitudes? What conditions ensure monotonicity of the shaping function or enable one to analytically derive peak amplitudes? What other techniques, such as concatenation of shaping functions, result in useful dynamics? And finally, what can be inferred about the physics and psychoacoustics of music from the underlying theoretical uniformity which is becoming increasingly apparent? It is hoped that further research will expand our information in these and other areas.

12. SUMMARY†

We have seen that waveshaping is a useful framework for both the generation of sound and the modeling of nonlinear synthesis systems. As a theoretical tool it posesses generality due to the central role played by the very general operation of function application. Yet it is analytically valuable because of the simplifications produced by the algebraization of seemingly complex trigonometric expressions and the natural and acoustically meaningful way in which waveshaping can be defined in terms of harmonic amplitudes.

As a practical method for musical synthesis, we have seen that waveshaping is quite cost effective and easily implementable. The ability to specify particular harmonic amplitudes is especially useful in sound generation. Analysis of the basic waveshaping algorithm has also led us to the construction of variations that are cheaper and others that are more flexible.

It remains to future research to determine the exact extent of the applicability of the ideas introduced here. To date the results of initial explorations have been quite encouraging. Natural instrument tones, as well as artificial sounds of musical interest, have been successfully synthesized and analyzed. We look forward to the further expansion and application of the newly developed but powerful techniques of waveshaping synthesis.

13. ACKNOWLEDGMENT

The author wishes to thank the members of the Center for Computer Research in Music and Acoustics and the Artificial Intelligence Laboratory at Stanford and the staff of the *Computer Music Journal* for their manifold assistances and encouragements in the course of his work. Among those who made special contributions to this research we would particularly like to thank James A. Moorer, Bill Schottstaedt, and Ken Shoemake of CCRMA, R. William Gosper and Hans Moravec of SAIL, and John Snell of CMJ.

We would also like to express our thanks to Daniel Arfib of C.N.R.S. Laboratoire de Mechanique et d'Acoustique, Marseilles, who independently originated and developed the application of waveshaping to the generation

of musical sounds, for many interesting discussions and insights.

Some of the results presented in this paper were obtained or suggested by interactions with MACSYMA [11], a sophisticated program for the symbolic manipulation of mathematical expressions, which is currently being developed by the Mathlab Group, Laboratory for Computer Science, Massachusetts Institute of Technology.

14. REFERENCES

[1] J. M. Chowning, "The Synthesis of Complex Audio Spectra by Means of Frequency Modulation," *J. Audio Eng. Soc.*, vol. 21, pp. 526–534 (1973 Sept.).

[2] J. A. Moorer, "The Synthesis of Complex Audio Spectra by Means of Discrete Summation Formulas," *J. Audio Eng. Soc.*, vol. 24, pp. 717–727 (1976 Nov.).

[3] M. R. Spiegel, *Mathematical Handbook of Formulas and Tables* (McGraw Hill, New York, NY 1968).

[4] M. Abramowitz and I. A. Stegun, *Handbook of Mathematical Functions*. (National Bureau of Standards, Boulder, CO, 1966 Aug.).

[5] R. A. Schaefer, "Electronic Musical Tone Production by Nonlinear Waveshaping," *J. Audio Eng. Soc.*, vol. 18, pp. 413–417 (1970 Aug.).

[6] C. Y. Suen, "Derivation of Harmonic Equations in Nonlinear Circuits," *J. Audio Eng. Soc. (Letters to the Editor)*, vol. 18, pp. 675–676 (1970 Dec.).

[7] D. Arfib, private communications, 1977 Oct.–Nov.

[8] M. Beeler, R. W. Gosper, and R. Schroeppel, "HAKMEM," Artificial Intelligence Memo 239, Massachusetts Institute of Technology, Cambridge, MA, 1972 Feb.

[9] A. Cuvelier, "Nouvelles Récherches sur la Synthèse de Sons en Temps Réel," in *Conferences des Journees D'Etudes*. (Editions Radio, Paris, 1976), pp. 87–105.

[10] B. Schottstaedt, "The Simulation of Natural Instrument Tones Using Frequency Modulation with a Complex Modulating Wave," *Computer Music J.*, vol. 1, pp. 46–50 (1977 Nov.).

[11] Mathlab Group, *MACSYMA Reference Manual*, Laboratory for Computer Science, Massachusetts Institute of Technology, Cambridge, MA, 1975 Nov.

Program 1. Computation of $f(x)$ by means of table lookup.

```
REAL PROCEDURE LOOKUP (REAL ARRAY F; REAL X);
BEGIN
COMMENT COMPUTE f(x) USING LINEAR INTERPOLATION BE-
  TWEEN POINTS;
  INTEGER L,I; REAL Lx,Ix,Rx;
  IF ABS(X)>1 THEN ERROR("X MUST BE IN SIGNED UNIT IN-
    TERVAL.")
  L ← LENGTH(F);            COMMENT F CONTAINS L ELE-
                              MENTS F[Ø] . . . F[L−1];
  Lx ← (X+1)*(L−1)/2;       COMMENT Lx LIES IN RANGE
                              [Ø L−1];
  Ix ← INT(Lx);             COMMENT TAKE INTEGER PART
                              OF Lx;
  Rx ← Lx−Ix;               COMMENT Rx CONTAINS THE
                              FRACTION PART OF Lx;
  IF Rx=Ø
    THEN RETURN             COMMENT EXACT ARRAY
      (F[Ix])                 INDEX;
    ELSE RETURN             COMMENT INTERPOLATE
      (Rx*F[Ix+1]+(1−Rx)      INDEX;
        *F[Ix]);
END;
```

Program 2. Computation of $T_n(x)$ by means of recursive formula.

```
REAL PROCEDURE T(INTEGER N; REAL X);
BEGIN
COMMENT COMPUTE Tn(X) BY MEANS OF Tn(X) = 2*X*Tn−1(X)
  − Tn−2(X)
  INTEGER I; REAL Tn,
    Tn1,Tn2;                COMMENT T[n](x), T[n−1](x)
                              AND T[n−2](x);
  Tn ← 1; Tn1 ← X;          COMMENT T[Ø] AND T[−1];
  FOR I←1 THRU N DO         COMMENT LOOP NEVER EXECUTES
                              FOR N=Ø;
  BEGIN
    Tn2 ← Tn1; Tn1          COMMENT PREPARE FOR COMPU-
      ← Tn;                   TATION OF Tn;
    Tn ← 2*X*Tn1−Tn2;       COMMENT RECURSIVE FORMULA
                              FOR NEXT Tn;
  END;
  RETURN(Tn);
END;
```

Program 3. Construction of shaping function from harmonic amplitudes.

```
PROCEDURE MAKESHAPE(REAL ARRAY F, Hk);
BEGIN
COMMENT COMPUTE f(x) USING HARMONIC AMPLITUDES IN Hk
  BY MEANS OF RECURSION; INTEGER Lf,Lh,I,K; REAL X,Tn,
  Tn1,Tn2;
  Lf ← LENGTH(F);           COMMENT GET TABLE SIZES;
    Lh ← LENGTH(Hk);
  FOR I←Ø THRU Lf−1 DO
  BEGIN
    X ← 2*I/(Lf−1)−1;       COMMENT X IS IN UNIT INTER-
                              VAL;
    F[I] ← Ø;               COMMENT CLEAR F ELEMENT FOR
                              SUM;
    Tn ← 1; Tn1 ← X;        COMMENT SET UP FOR RECURSION;
    FOR K←Ø THRU Lh DO
    BEGIN
      F[I] ← F[I]
        +Hk[K]*Tn;          COMMENT ADD IN WEIGHTED Tn;
      Tn2 ← Tn1;            COMMENT PREPARE TO COMPUTE
        Tn1 ← Tn;             NEXT Tn;
      Tn ← 2*X*Tn1          COMMENT RECURSIVE FORMULA
        −Tn2;                 FOR Tn+1;
    END;
  END;
END;
```

Program 4. Subprogram for computing dynamic spectra.

```
BEGIN SUBPROGRAM
COMMENT SUBPROGRAM ENVIRONMENT FOR CONSTRUCTING AND
  EVALUATING Hk(a) FUNCTIONS;
EXTERNAL INTEGER M;         COMMENT MAXIMUM HARMONIC
                              NUMBER;
EXTERNAL REAL ARRAY         COMMENT STEADY STATE HAR-
  Hk[Ø:M];                    MONIC AMPLITUDES;
INTEGER PROCEDURE COSIZE(INTEGER N);
BEGIN
COMMENT NUMBER OF ELEMENTS IN COEFFICIENT ARRAY FOR N
  HARMONICS;
  RETURN(INT((N+2) ↑ 2/4)−1);
END;
```

Program 4 (continued)

```
REAL ARRAY COEFF[Ø:COSIZE(M)];
  COMMENT ARRAY OF COEFFICIENTS FOR HARMONIC POLYNO-
    MIALS. COEFF IS STRUCTURED WITH THE COEFFS FOR HM
    FIRST, IN ORDER OF INCREASING POWERS OF "a", THEN
    HM-1 NEXT AND SO ON DOWN TO THOSE FOR HØ.;
INTEGER PROCEDURE BASEADDR(INTEGER K);
BEGIN
COMMENT BASE ADDRESS OF KTH SET OF VALUES IN COEFF;
  RETURN(COSIZE(M)-COSIZE(M-K));
END;
REAL PROCEDURE H(INTEGER K; REAL A);
BEGIN
COMMENT EVALUATE HARMONIC POLYNOMIAL AT A;
  INTEGER I,MAXI; REAL Z, A2;
  Z←Ø; A2←A*A;
  MAXI←INT((M-K)/2);
  FOR I←Ø THRU MAXI DO Z←Z*A2+COEFF[BASEADDR(K)+
    MAXI-I];
  RETURN(Z*A ↑ K);
END;
PROCEDURE MAKECOEFF;
BEGIN
COMMENT FILL COEFFICIENT ARRAY AS DETERMINED BY STEADY
  STATE VALUES IN Hk;
  REAL ARRAY D[Ø:M];
  INTEGER P,K,I,S;
  FOR P ← Ø THRU M DO
  BEGIN
  COMMENT COMPUTE D VALUES;
    S ← 1; D[P] ← Ø; COMMENT INITIALIZE ALTERNATING SIGN,
      ARRAY ELEMENT;
    FOR I ← Ø THRU INT((M-P)/2) DO
    BEGIN
      D[P] ← D[P] + S * (BINOMIAL(P+I,I) + BINOMIAL
        (P+I-1,I-1)) * Hk[P+2*I]; S ← -S;
    END;
  END;
  FOR K ← Ø THRU M DO
    FOR I ← Ø THRU INT((M-K)/2) DO
      COEFF[BASEADDR(K)+I] ← BINOMIAL(K+2*I,I) *
        D[K+2*I];
END;
END SUBPROGRAM;
```

Program 5. Computation of normalized output.

```
REAL PROCEDURE NORMALIZEOUTPUT( REAL ARRAY F, N; REAL
  X, A);
BEGIN
COMMENT COMPUTE NORMALIZED OUTPUT f(ax)/N(a);
  REAL Z;                      COMMENT NORMALIZATION VARI-
                                 ABLE;
  Z ← LOOKUP(N,2*A-1);         COMMENT A MAPPED INTO
                                 SIGNED UNIT INTERVAL;
  IF Z=Ø
    THEN RETURN( LOOKUP        COMMENT ZERO, DON'T NOR-
      (F,A*X))                   MALIZE;
    ELSE RETURN(LOOKUP         COMMENT NORMALIZE;
      (F,A*X)/Z);
END;
```

Program 6. Construction of peak normalization array.

```
PROCEDURE MAKEPEAKNORM(REAL ARRAY F, N);
BEGIN
COMMENT CONSTRUCT THE N ARRAY THAT PEAK NORMALIZES
  THE FUNCTION IN THE F ARRAY;
  INTEGER L,I;
  L ← LENGTH(N);
  N[Ø] ← ABS(LOOKUP(F,Ø));
  FOR I ← 1 THRU L-1 DO
  BEGIN
    REAL X;
    X ← I/(L-1);               COMMENT X IS IN THE UNSIGNED
                                 UNIT INTERVAL;
  COMMENT TAKE 3 WAY MAXIMUM OF PREVIOUS VALUE AND
    SYMMETRIC POINTS;
  N[I] ← MAX(F[I-1], ABS(LOOKUP(F,X)), ABS(LOOKUP(F,
    -X)) );
  END;
END;
```

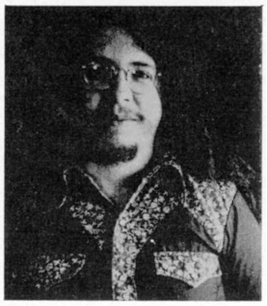
Program 7. Construction of shaping functions for phase quadrature.

```
PROCEDURE MAKEFxFy(REAL ARRAY H, Phi, Fx, Fy);
BEGIN
COMMENT CONSTRUCT PHASE QUADRATURE f (Fx) AND BAR f
  (Fy) GIVEN H AND Phi;
  INTEGER I,K,L,N;             COMMENT INDICES, fn LENGTH &
                                 MAX k;
  REAL Tk,Tk1,Tk2;             COMMENT T RECURRENCE VARI-
                                 ABLES;
  REAL Uk,Uk1,Uk2;             COMMENT U RECURRENCE VARI-
                                 ABLES;
  REAL X;                      COMMENT POLYNOMIAL ARGU-
                                 MENT;
  N ← LENGTH(H)-1;             COMMENT GET MAX HARMONIC
                                 NUMBER;
  REAL ARRAY CPhi,SPhi         COMMENT COS(Phi) AND SIN(Phi)
    [Ø:N];                       TABLES;
  FOR K ← Ø THRU N DO          COMMENT FILL CPhi AND SPhi
    BEGIN                        TABLES;
    CPhi[K] ← H[k]*cos(Phi[K]);
    SPhi[K] ← H[k]*SIN(Phi[K]);
  END;
  L ← LENGTH(Fx)-1;            COMMENT GET MAX INDEX OF
                                 SHAPING TABLES;
  FOR I ← Ø THRU L DO          COMMENT FILL SHAPING TABLES;
    BEGIN
    X ← 2*I/L-1;               COMMENT MAP I INTO SIGNED
                                 UNIT INTERVAL;
    Tk ← 1; Uk ← Ø;            COMMENT INITIALIZE FOR K=Ø;
    Tk1 ← X; Uk1 ← -1;         COMMENT INITIALIZE FOR k=
                                 -1;
    Fx[I] ← Ø; Fy[I] ← Ø;      COMMENT PREPARE FOR SUMMA-
                                 TION;
    FOR K ← Ø THRU N           COMMENT FILL ARRAYS;
      DO BEGIN
      Fx[I] ← Fx[I]+CPhi       COMMENT ADD IN WEIGHTED
        [K]*Tk;                  VALUES;
      Fy[I] ← Fy[I]+
        SPhi[K]*Uk;
      Tk2 ← Tk1; Tk1           COMMENT PREPARE TO COMPUTE
        ← Tk;                    NEXT Tk;
      Tk ← 2*X*Tk1             COMMENT RECURSIVE FORMULA
        -Tk2;                    FOR Tk + 1;
      Uk2 ← Uk1; Uk1 ←         COMMENT DITTO FOR Uk;
        Uk;
      Uk ← 2*X*Uk1-Uk2;
    END;
  END;
END;
```

Program 8. Complete phase quadrature waveshaping synthesis algorithm.

```
REAL PROCEDURE PQWS( REAL ARRAY Fx, Fy; REAL X, Y, A);
BEGIN
COMMENT PHASE QUADRATURE WAVE SHAPING ALGORITHM —
  X & Y ARE COS & SIN;
  REAL Z;                      COMMENT OUTPUT VARIABLE;
  Z ← LOOKUP(Fx,A*X)+A*Y*LOOKUP(Fy,A*X);
  RETURN( Z);
END;
```

THE AUTHOR



Marc Le Brun was born in 1952 in Phoenix, Arizona. He is currently a mathematical systems programmer at the Stanford Artificial Intelligence Laboratory and a guest researcher at the Center for Computer Research in Music and Acoustics.