Andrew Horner, James Beauchamp, and Lippold Haken
CERL Sound Group, Computer Music Project, and
Center for Complex Systems Research
University of Illinois at Urbana-Champaign
Urbana, Illinois 61801 USA
horner@cmp-nxt.music.uiuc.edu

Machine Tongues XVI: Genetic Algorithms and Their Application to FM Matching Synthesis

Historically, frequency modulation (FM) synthesis has required trial and error to create emulations of natural sounds. This article presents a genetic-algorithm-based technique which determines optimized parameters for reconstruction through FM synthesis of a sound having harmonic partials. We obtain the best results by using invariant modulation indices and a multiple carrier formant FM synthesis model. We present our results along with extensions and generalizations of the technique.

Background

FM synthesis is a very efficient, though not always easily controlled technique for generating interesting sounds. A small number of parameters can be varied to produce a wide range of sound timbres. In general, FM parameter matching for systematic reconstruction of an arbitrary musical instrument tone has remained an open question. However, a number of ad hoc techniques have been introduced and will be reviewed here.

Emulation of acoustic musical instruments via trial-and-error FM synthesis parameters dates back to John Chowning's original paper on the technique (Chowning 1973). In addition to introducing the application of FM in music synthesis, Chowning gave general parameter settings appropriate to various classes of instruments. His observations were based on the simulation of various properties of those instruments. In a later study, Chowning also designed an instrument simulating a singing soprano voice (Chowning 1980). Dexter Morrill's study of FM trumpet tones (Morrill 1977) followed Chowning's lead in trying to determine the FM parameters based on detailed knowledge of the behavior of the trumpet sounds.

An initial attempt to automate FM matching can be found in Justice's paper on analytic FM matching (Justice 1979). Justice outlined a procedure based on the discrete Hilbert transform to decompose a signal into parameters for a single carrier FM instrument with nested modulators. The basic idea of this procedure is that this analysis attempts to find a coarse match between the FM signal and the original signal. At that point the user can "tweak" the parameters to refine the match. However, Justice does not address the question of whether many "real-world" signals fit this particular model. Indeed, the signals he gave as examples were themselves FM-generated, rather than being signals taken from acoustic musical instruments. Nonetheless, the work is of theoretical interest and was a basis for two later studies that did apply the technique to some acoustic instrument tones (Payne 1987; Delprat, Guillemain, and Kronland-Martinet 1990). These methods, however, still required a large amount of data processing.

In contrast to these analytic FM matching techniques which work in the time domain, a frequency domain approach was developed by one of us (Beauchamp 1982) in an attempt to find FM parameters as part of a larger study involving spectral centroid matching in nonlinear synthesis. With this method, the FM modulation index of a single modulator-carrier model is manipulated to track the time-variant spectral centroid of the signal. Although the level of control was too coarse to provide a good perceptual match to the original sound, the technique was notable as an early attempt to perform an automated spectral match.

The FM matching synthesis technique presented in this article finds "optimized" parameters for FM synthesis of an analyzed sound via a genetic algo-

Computer Music Journal, 17:4, pp. 17–29, Winter 1993,
© 1993 Massachusetts Institute of Technology.

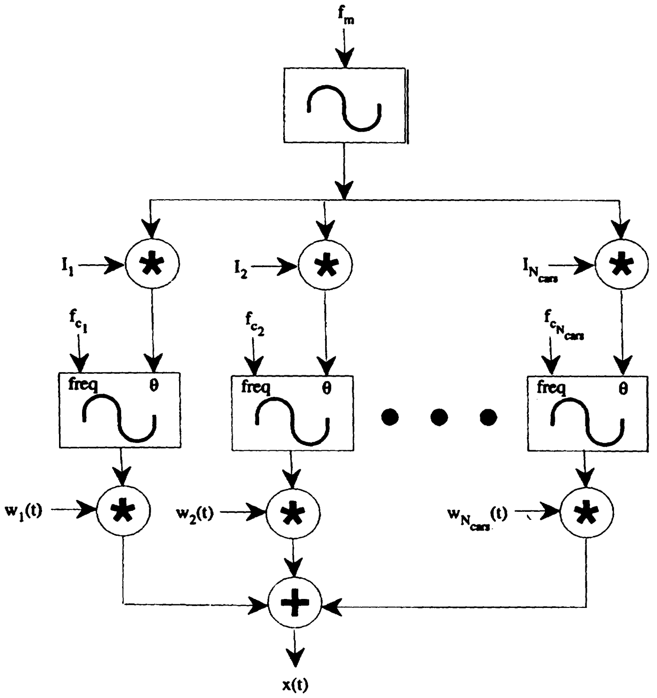


Fig. 1. FM synthesis model.

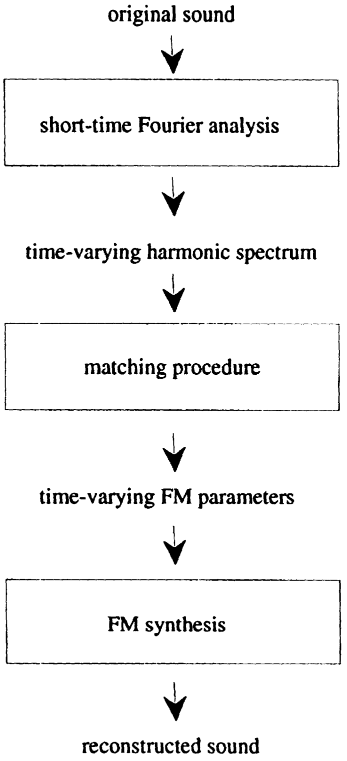

original sound
short-time Fourier analysis
time-varying harmonic spectrum
matching procedure
time-varying FM parameters
FM synthesis
reconstructed sound

Fig. 2. FM matching synthesis overview.

rithm. Like several previous investigators (e.g., Morrill 1977; Chowning 1980), we use a synthesis model consisting of a single modulator and multiple parallel carriers, as shown in Fig. 1. The parameters matched consist of the frequencies and amplitudes of the modulator and carriers. Currently, the matching synthesis technique works for an arbitrary tone having harmonic partials, although we expect that it can be extended to handle some classes of sounds composed of inharmonic partials.

In our implementation, FM matching and verification consists of three main steps: short-time spectral analysis, parameter matching using a genetic algorithm, and sound synthesis from an FM model, as illustrated in Fig. 2. In the analysis step, a short-time Fourier analysis algorithm is applied to the sound to be matched. The result is a set of discrete-time spectra, which we interpret as partials varying in time and frequency. By determining a set of time-varying FM parameters that yield spectra best matching the original spectra, we are able to reconstruct the sound using FM synthesis. The principal contribution of this article is a method for efficient and effective determination of the FM parameters.

Genetic algorithms (Holland 1975; Goldberg 1989) are evolution-based optimization techniques that have been applied to a wide variety of problems from stack filter design (Chu 1989) to computer-assisted composition (Horner and Goldberg 1991). Unlike most traditional optimization techniques, genetic algorithms (GAs) do not rely on a particular problem structure or problem-specific knowledge. The general effectiveness of GAs as optimization procedures which are blind to the peculiarities of any particular problem accounts in no small part for their growing popularity. The task of finding FM parameters to match musical tones is typical of problems that defy traditional optimization, yet are suited to genetic algorithm solution.

FM Synthesis

The general equation for single-modulator/single-carrier frequency modulation (or, more accurately, *phase modulation*) synthesis can be written

$$x(t) = \omega(t) \sin(2\pi \int f_c(t)dt + I(t) \sin(2\pi \int f_m(t)dt)), \quad (1a)$$

where $w(t)$ is the time-varying amplitude (sometimes called a *weight* in this article), f_c is the *carrier frequency*, f_m is the *modulation frequency*, and I is the *modulation index*, each of which could be time-varying. This equation can be expanded in terms of Bessel functions as follows:

$$x(t) = w(t) \sum_{k=-\infty}^{\infty} J_k(I(t)) \sin(2\pi \int (f_c(t) + k f_m(t)) \, dt), \quad (1b)$$

where $J_k()$ is the Bessel function of the first kind, kth order. Note that $J_{-k}() = (-1)^k J_k()$.

Due to the nature of Bessel functions, the bandwidth of the waveform's spectrum generally increases as the index I increases, although a great deal of oscillation of the individual partial amplitudes accompanies this growth. Indeed, some of the partial amplitudes may be negative for particular values of I, corresponding to phase inversion of those partials. As a result, when two FM signals are summed, coincident partials of the two signals will either add or partially cancel one another. Within a single FM signal, partials having "negative frequencies" can add or partially cancel partials having positive frequencies. Such positive and negative reinforcements (to borrow a term from acoustics) play a crucial role in sculpting spectra and must be taken into account during the FM matching process.

Finally, if f_c and f_m are related by a rational multiplier, a harmonic tone results. When f_c is an integer multiple of f_m, we have a special case of this result, and the fundamental frequency will be $f_1 = f_m$. This case is the one used in our current FM matching research based on the genetic algorithm.

Letting, $f_c = n\, f_m = n\, f_1$ in Eq. (1b) the expression for a single carrier output becomes

$$x(t) = w(t) \sum_{k=-\infty}^{\infty} J_k(I(t)) \sin(2\pi(n+ k) \int f_1(t)dt) \quad (2a)$$

$$= w(t) \sum_{k=1}^{\infty} [J_{(k-n)}(I(t)) - J_{-(k+n)}(I(t))]\sin(2\pi k \int f_1(t)dt) \quad (2b)$$

$$= w(t) \sum_{k=1}^{\infty} a_k(I(t)) \sin(2\pi k \int f_1(t)dt). \quad (2c)$$

We have dropped the zero frequency term because we believe its effect is inaudible in most situations. We see that for each harmonic the amplitude can be expressed as the difference of two Bessel functions evaluated at the index. Frequently the second Bessel function term will be nearly zero. Also, each amplitude a_k can have either a positive or negative sign, a fact that we will take into account in our matching procedure, which we discuss below.

If we combine several carrier outputs, the total output is

$$x(t) = \sum_{j=1}^{N_{cars}} w_j(t) \sum_{k=1}^{\infty} a_{k,j}(I_j(t)) \sin(2\pi k \int f_1(t)dt) \quad (2d)$$

where $w_j(t)$ is the time-varying weight (amplitude envelope) applied to the jth carrier, $a_{k,j}(I_j(t)) = J_{(k-n_j)}(I_j(t)) - J_{-(k+n_j)}(I_j(t))$ is the kth harmonic amplitude of the jth carrier, and $n_j = f_{cj}/f_m$ is the ratio between the jth carrier frequency f_{cj} and the common modulator frequency f_m. This leads to

$$x(t) = \sum_{k=1}^{\infty} b'_k(t) \sin(2\pi k \int f_1(t)dt) \quad (2e)$$

where

$$b'_k(t) = \sum_{k=1}^{\infty} w_j(t)\, a_{k,j}(I_j(t)) \quad (2f)$$

is the time-varying amplitude of the kth harmonic in the FM-synthesized signal.

Short-Time Spectrum Analysis

Matching of an FM-synthesized signal to an original musical sound signal is facilitated by working in the frequency domain and matching the sound spectra of the original and synthesized signals. The basic assumption we make about any original signal is that it can be represented by a sum of sine waves with time-varying amplitudes and frequencies; that is,

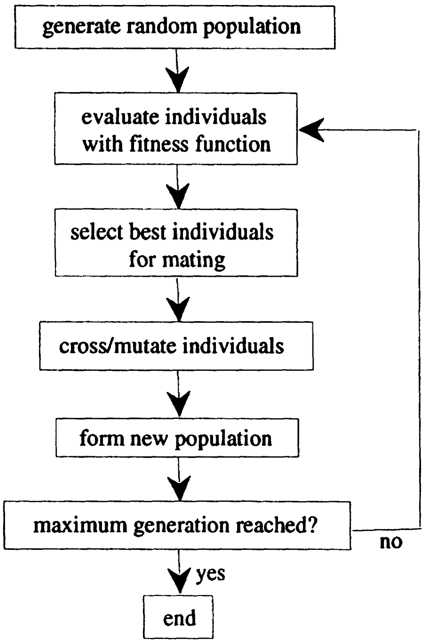
$$y(t) = \sum_{k=1}^{N_{hars}} b_k(t) \sin(2p \int f_k(t)dt + q_k), \qquad (3a)$$

where $b_k(t)$ and $f_k(t)$ are the time-varying amplitude and frequency of the kth partial in the original signal. Note that in our matching procedure we will ignore the starting phases θ_k as having an inaudible effect in most situations. A further restriction that we employ in this article is that the sound is harmonic, so that

$$f_k(t) = k\, f_1(t). \qquad (3b)$$

We use two different techniques to analyze a sound in order to estimate its amplitudes and frequencies. Our usual method is a fixed filter bank approach where bandpass filters are centered on the harmonics of an "analysis frequency" which approximates the mean of $f_1(t)$ (Maher 1989). In this method, the filter outputs give real and imaginary parts which are converted into amplitudes and phases by right triangle solution. The frequencies are then computed as the derivatives (or finite differences) of the phases. This method fails, however, if $f_1(t)$ frequency deviations are so great that upper harmonic frequencies swing on the edges or outside the ranges of the filters. In this case (e.g., for substantial vibrato or portamento situations), we use an extension of the McAulay-Quatieri analysis technique (McAulay and Quatieri 1986), which is capable of tracking severe changes of pitch. By itself, the "MQ" method computes amplitudes and frequencies on the basis of finding *peaks* in a spectrum computed with a fixed-fundamental fast Fourier transformation (FFT). However, since this method is not inherently restricted to harmonics, harmonic frequencies must be "interpreted" from the data for the peaks. We do this by first estimating the fundamental frequency as a function of time from the MQ spectral data, then sorting this data into the appropriate harmonic bins (Maher 1989).

Fig. 3. Genetic algorithm overview.


generate random population
evaluate individuals with fitness function
select best individuals for mating
cross/mutate individuals
form new population
maximum generation reached?
no
yes
end

Genetic Algorithms

Genetic algorithms (GAs) are combinatorial optimization techniques modeled on the idea of evolving solutions to problems through processes analogous to natural selection and breeding. These evolutionary processes guide the search of the problem environment using populations of candidate solutions.

The procedure generally begins with a random initial population of *candidate solutions*, which are also referred to as *individuals* in GA terminology. Genetic operators are then applied iteratively to produce new populations of individuals over successive generations, as illustrated in Fig. 3. The user supplies a problem-specific *fitness function* that evaluates a given candidate solution when it enters the population. The best candidate solution evaluated by this fitness function over all the generations of the run is subsequently chosen as the answer to the problem. As with most other iterative techniques, no guarantees are made as to the optimality of this answer, but it is generally a very good solution.

Along with the fitness function, the user must decide on an encoding for candidate solutions. Generally GAs work with binary encodings. For integer parameters, such an encoding simply entails using enough bits to capture the range of the parameter's values. For instance, numbers in the range [10, 25] can be encoded by a 4-bit binary number if 10 is added to their value during decoding. Real valued numbers are encoded similarly, but by using scaling in addition to translation. As an example, a 4-bit representation of numbers in [0, 1) would entail scaling the binary integers by 0.0625 during decoding.

The motivation for going to the trouble of specifying a binary encoding is that the genetic algorithm can then apply its genetic operators on this binary encoding without needing to know explicitly which bits belong to which parameters. Only the user-supplied fitness function typically makes use of this information in decoding the parameters and then evaluating how well they perform on the problem. The GA takes advantage of the uniform binary encoding by freely mixing and matching substrings of candidate solutions, as we will soon see.

The standard GA uses *selection*, *crossover*, and *mutation* as its genetic operators. Selection propagates the best individuals of a population through succeeding generations and weeds out those less fit. Crossover is the GA counterpart to sexual breeding. It should be noted, however, that it most closely resembles two-parent breeding as practiced by simple organisms. Crossover mixes and matches parts of candidate solutions in search of superior combinations of solution building blocks. Finally, mutation is a hill-climbing operator that drives exploration of the local fitness landscape around a candidate solution. Mutation randomly alters a small number of bits within an individual. Selection, crossover, and mutation work together efficiently to explore the solution space by propagating and recombining the best individuals in each generation.

Several different selection schemes exist, but one that is particularly effective is binary tournament selection. In this scheme, members of the population are randomly paired up. For each of the pairs, the individual with the best fitness is selected to participate in the new population. A second random pairing among all the individuals of the old population is performed to select a second group of participants. This ensures that the new population will contain the same number of individuals as the old population.

Also note that binary tournament selection guarantees that the best individual in the old population will show up twice in the new population. Conversely, the worst individual will not be propagated to the new population at all. Other individuals will propagate zero to two individuals, depending on the particular pairings they end up in. However, better than average individuals will constitute the majority of the new population as desired.

As a simple example, suppose we have a population of size four with the individuals

Individual Number	Individual Bitstring	Individual Fitness
1	01101	17
2	11000	57
3	01000	6
4	10011	36

Suppose that individuals 1 and 2 are paired up, leaving individuals 3 and 4 to form the other pair. In the first pairing, individual 2 will be selected, while individual 4 will win the other pairing. If the second set of pairings put individual 2 with 4, and individual 3 with 1, the propagated group will be

Individual Number	Individual Bitstring	Individual Fitness
2	11000	57
4	10011	36
2	11000	57
1	01101	17

The average individual fitness of our initial group was 29. In the propagated group we expect the average to increase, and it does with a value of 41.75.

The individuals propagated by the selection operation either mate via crossover to produce offspring for the new population, or simply get placed in the new population uncrossed. If selected for crossover, individuals are randomly paired up, and substrings of their bitstrings are swapped at a random crossing point. For instance, with a crossing point after bit 3, and individuals

```
000   00000 and
111   11111
```

we get

```
000   11111 and
111   00000.
```

Generally, about 60 percent of individuals will mate while the other 40 percent are propagated without crossover. Since crossover is a potentially disruptive operation (i.e., it may result in inferior offspring), leaving some individuals uncrossed ensures that there will generally be above average individuals in the new population. On the other hand, crossover allows for the discovery of potentially useful combinations of bit patterns that would not otherwise occur.

After selection and crossover, mutation is the final genetic operator applied before the new population is complete. Within each propagated individual, each bit is given a small chance of being changed. Generally the probability of this change is on the order of the reciprocal of the bitstring length. This means that one bit of the individual will be changed in expectation. This allows solutions neighboring those already seen to be explored within the Hamming space of the encoded problem.

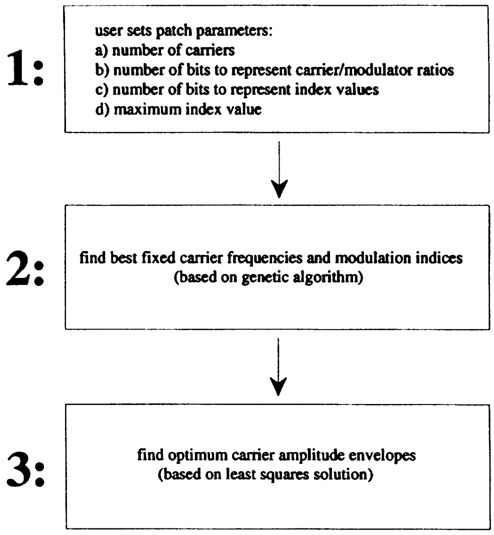

1:
user sets patch parameters:
a) number of carriers
b) number of bits to represent carrier/modulator ratios
c) number of bits to represent index values
d) maximum index value
2:
find best fixed carrier frequencies and modulation indices
(based on genetic algorithm)
3:
find optimum carrier amplitude envelopes
(based on least squares solution)

Fig. 4. Three-step matching procedure.

Matching Procedure

The matching procedure to determine the FM synthesis parameters consists of three steps, illustrated in Fig. 4. It turns out that the second step, using the genetic algorithm to determine the constant carrier frequencies and modulation indices, is the most crucial. This step, however, is dependent on the method used in finding the optimum carrier amplitudes, which comprises the third step. We shall first discuss the general implications of steps 1 and 2, but we will put off a complete discussion of step 2 until the procedure for step 3 is described.

As the first step, the user specifies the FM patch to be used in synthesis, which in the current study is restricted to the case of a single modulator controlling a group of carriers. Each of these carriers has a frequency harmonically related to the modulator frequency, an individual fixed modulation index, and a time-varying amplitude envelope (see Fig. 1). Other possible patch configurations are discussed below. With our current program, the patch parameters the user must specify are the number of carriers, N_{cars}, the number of bits to represent carrier-to-modulator ratios, the number of bits to represent index values, and the maximum index value allowed. The number of bits used to represent the carrier-to-modulator ratios implicitly determines the integer value of the highest potential carrier-to-modulator frequency ratio, n_{max}. The number of carriers influences the closeness of the match (the more carriers, the better) and the amount of data reduction and/or ease of real-time implementation (the less carriers, the better). Obviously, the number of carriers should be substantially less than the number of harmonics in the sound being reconstructed. Otherwise, additive synthesis would be the preferred method, since it would be just as efficient and much more straightforward to implement. With FM, however, we have, in effect, a control on the trade-off between accuracy and efficiency.

Next, in step 2 the fixed FM parameters are determined using a genetic algorithm. These include the carrier-to-modulator frequency ratios and, perhaps surprisingly, invariant modulation indices.

Why should we restrict our attention to FM synthesis using fixed indices? Well, first of all, if we allow the indices to vary, then we are forced to run a computationally expensive nested optimization routine to determine them. Even if we accept this penalty, there is a further problem with time-varying indices. In initial experiments, a nested genetic algorithm designed to determine time-varying modulation indices and carrier amplitudes not only consumed inordinate amounts of computer time, it also tended to produce discontinuous index functions. Moreover, while matches on both sides of discontinuities were typically very good, the associated FM parameters were usually radically different. Interpolation of these parameters across the discontinuity boundaries resulted in wildly changing spectra, generally heard in the resulting synthetic sounds as "pops." An initial attempt to solve this problem by limiting the amount of allowed index change from one envelope point to the next resulted in degraded matches. As an alternative, regions of small absolute deviations were imposed, which surprisingly resulted in good-sounding matches. Almost as surprisingly, we found that using certain fixed indices achieved results which sounded just as good. Therefore, we conclude that it is best to reduce the problem to the determination of a series of fixed indices $I_1, \ldots, I_{N_{cars}}$.

Along with the fixed modulation indices, we must determine constant carrier-to-modulator frequency ratios. Because we currently restrict our matching to sounds with harmonic partials, we must set the carrier and modulator frequencies to multiples of the fundamental frequency to reflect this. Many configurations of carriers and modulators are possible (LeBrun 1977), but the basic idea remains the same. In our model, a harmonic spectrum results from setting a common modulator frequency to the desired fundamental and the carrier frequencies to integer multiples of the fundamental.

$$f_m = f_1; \quad f_{c_1} = n_1 f_1; \quad f_{c_2} = n_2 f_1; \quad \ldots \quad f_{cN_{cars}} = n_{N_{cars}} f_1, \tag{4}$$

where $0 \le n_j \le n_{max} \le N_{hars}$, and $1 \le j \le N_{cars}$. Note that N_{cars}, the number of carriers, and n_{max}, the maximum potential carrier-to-modulator frequency ratio, are specified by the user at the outset of the matching process, at least implicitly. Also note that we include the possibility of a zero frequency carrier, which, it turns out, generates only odd harmonics of the fundamental. The integers n_1, n_2, etc., are the carrier-to-modulator frequency ratios which, in addition to the modulation indices, are determined by the search procedure of step 2.

Finally, in step 3 we determine the optimum amplitude envelopes by straightforward calculation. With constant modulation indices, we essentially have a set of fixed "basis spectra." Using these basis spectra and the discrete-time spectra of the original sound, we form a system of linear equations represented by the matrix equation

$$AW \approx B \tag{5a}$$

As Fig. 5 shows, the matrix A contains the FM basis spectra stored as a series of columns, with one column for each carrier spectrum; the matrix W contains the unknown carrier amplitude weights, corresponding to time samples of the (as yet undetermined) carrier envelopes for each time frame of the analysis, arranged in a series of columns; and the matrix B contains successive frames of the original discrete-time spectra. This system of equations is of the form

$$\sum_{j=1}^{N_{cars}} a_{k,j} w_{j,r} \approx b_{k,r}, \quad 1 \le k \le N_{hars} \tag{5b}$$

where $a_{k,j}$ is the time-fixed amplitude of the kth harmonic due to the jth carrier, $w_{j,r}$ is the envelope weight for the jth carrier at the rth time frame, and $b_{k,r}$ is the amplitude of the kth harmonic of the original spectrum at the rth time frame. If the number of carriers (N_{cars}) is equal to the number of harmonics (N_{hars}) of the original sound, Eq. (5a,b) can be solved exactly by direct matrix solution, so a perfect solution results. What we want, however, is a reduced set of carriers. For this case, we can determine a best solution in the least-squares sense. This is tantamount to determining the $\{w_{j,r}\}$ that minimize

$$\sum_{k=1}^{N_{cars}} \sum_{j=1}^{N_{cars}} (a_{k,j} w_{j,r} - b_{k,r})^2 \tag{5c}$$

for each time frame r.

Fortunately, very efficient algorithms exist for direct least-squared solution of Eq. (5a), given the matrices A and B (Press et al. 1989). Thus, for a given set of basis spectra (i.e., those generated by the FM parameters), the computation of their amplitude envelopes is a straightforward process.

Now let us look at step 2 in more detail. The genetic algorithm optimization routine selectively

Fig. 5. Matrix representation of Eq. (5a).

$$\begin{bmatrix} a_{1,1} & \cdots & a_{1,N_{cars}} \\ a_{2,1} & \cdots & a_{2,N_{cars}} \\ \cdot & & \\ \cdot & & \\ a_{N_{hars},1} & \cdots & a_{N_{hars},N_{cars}} \end{bmatrix} \mathbf{X} \begin{bmatrix} w_{1,1} & \cdots & w_{1,N_{frames}} \\ w_{2,1} & \cdots & w_{2,N_{frames}} \\ \cdot & & \\ \cdot & & \\ w_{N_{cars},1} & \cdots & w_{N_{cars},N_{frames}} \end{bmatrix} \approx \begin{bmatrix} b_{1,1} & \cdots & b_{1,N_{frames}} \\ b_{2,1} & \cdots & b_{2,N_{frames}} \\ \cdot & & \\ \cdot & & \\ b_{N_{hars},1} & \cdot\cdot & b_{N_{hars},N_{frames}} \end{bmatrix}$$

Fig. 5.

Fig. 6. An individual and its FM parameters. The numbers of bits typically used in the encoding are shown beneath the individual's parameters.

n_1	I_1	n_2	I_2	. . .	$n_{N_{cars}}$	$I_{N_{cars}}$	s_1	s_2	. . .	$s_{N_{hars}}$
4	7	4	7		4	7	1	1		1

Fig. 6.

searches for the best set of FM parameters (i.e., the set of carrier indices and carrier-to-modulator frequency ratios), by generating candidates that it subsequently propagates and recombines on the basis of their fitness values. While performing this search, the GA continually calls upon the least-squares algorithm to determine the carrier amplitude weights which minimize the sum-squared difference between the reconstructed spectrum and original spectrum (see Eq. 5c), averaged over a number of discrete spectra within the sound. It then uses the resulting average difference as the fitness value for each candidate in order to guide the subsequent search.

The fitness function for a given individual, where we are trying to minimize, is thus given by

$$\sum_{i=1}^{N_{cars}} \left[\sum_{i=1}^{N_{cars}} (b_k(t_i) - b'_k(t_i))^2 \Big/ \sum_{k=1}^{N_{cars}} b_k^2(t_i) \right]^{1/2}, \quad (5d)$$

where the t_i are selected time values within the duration of the sound being matched, N_{frames} is the number of time values selected, and b_k, b'_k, and N_{hars} are defined by Eqs. (2f and 3a).

The computational cost of the search procedure can be held down considerably by restricting the time average to a limited number of representative spectra from the sound being matched. Judicious choice of spectra from the original sound is important in achieving a reasonable FM approximation. For example, spectra in the attack portion of a sound are very good choices, as the attack is a perceptually critical and a fast-changing portion of the tone (Clark et al. 1963; Berger 1963; Grey and Moorer 1977). After considerable experimentation, we typically pick 10 spectra equally spaced in time between the beginning of the sound and the point where the highest amplitude of the sound occurs. Sounds that radically change character during their decay obviously require a different strategy for selecting the time points.

The binary encoding of FM parameters within individuals is fairly straightforward. For each carrier, a carrier-to-modulator frequency ratio n_j and a modulation index I_j must be determined. Figure 6 illustrates how many bits are typically used to represent each parameter. The reason for the trailing s_k parameters will be explained in the next section. Note that with 4 bits allocated to encode the frequency ratios, n_{max} (mentioned after Eq. 4) will have a value of 15. Adding another bit to this allocation would increase n_{max} to 31. The modulation indices, on the other hand, are real-valued parameters. With indices in the range [0.0, 10.0] and a 7-bit encoding of the indices, the resolution will be about 0.08.

A simple genetic algorithm with binary tournament selection, one-point crossover, and mutation has been employed in our spectral matching work. The encoding depicted in Fig. 6 is viewed as a single bitstring by the GA; thus mixing and matching of bits during crossover is not restricted to parameter boundaries. The GA does not even know such parameter boundaries exist, but it is able to determine from the fitness evaluation when a good combination of parameters has been found.

Additional Considerations

We have determined the fixed carrier-to-modulator frequency ratios and modulation indices and the time-varying carrier amplitude envelopes. Do we need to determine anything else? It turns out that some additional parameters must be taken into account for both the matching process and the synthesis step.

The matrix equation of Eq. (5a) is a little too simplistic. This is because the signs of many of the harmonic amplitudes in FM-generated spectra (the $a_{k,j}$ of Eq. 2d) are negative, reflecting a particular phase distribution for these harmonics. Therefore, our equation $AW \approx B$ must be modified to allow for this. A simple method of doing this is to let the signs of the original spectra amplitudes be either positive or negative. (Remember that we are assuming that the ear is impervious to phase.) We can express this more formally. First, we let D be a diagonal square matrix where each dimension is equal to the number of harmonics. The diagonal elements are unknown but must have values of either plus or minus 1. We then construct the matrix equation

$$AW \approx DB. \tag{6}$$

Minimization of error then involves finding both W and D. Enumeration is not a good alternative in determining the signs of the D elements, since typically there will be 20 to 30 harmonics, and thus at least $2^{20} \approx 1{,}000{,}000$ possibilities. However, we can simply make each sign a 1-bit parameter that the genetic algorithm determines. These are the final N_{hars} parameters of our encoding in Fig. 6. With $C = DB$ for some given candidate D, we can then use least-squares to minimize $AX \approx C$. Thus, the genetic algorithm will be trying to find the set of modulation indices, carrier ratios, and amplitude signs that make the best match.

On another point, the fundamental frequency deviation $f_d(t)$ found by the short-time Fourier analysis normally is used to vary the fundamental frequency of the FM tone for more natural results. Since the modulator and carrier frequencies are locked to the fundamental, they must be varied in proportion to this deviation;

$$f_m(t) = f_1 + f_d(t)\ ;$$
$$f_{c_i}(t) = n_i\,[f_1 + f_d(t)]. \tag{7}$$

Matching Results

So how many carriers are needed for a good match? How many for an exact match? The latter question turns out to be trivial; if the number of carriers equals the number of harmonics of the original sound, then the modulation indexes can be set to zero and an exact match can be made through the resulting additive synthesis. The first question has to be answered more empirically, however. In the matches tried to date, four or five carriers seem to be quite adequate in achieving good results for even more difficult to match sounds.

The matches performed to date have been based on the sounds of a trumpet, tenor voice, guitar, oboe, bassoon, clarinet, trombone, and viola. Detailed results from the first three of these instruments will be presented here. The trumpet was remotely approximated when only one carrier was used in the FM matching. On the other hand, the one-carrier match to the tenor voice was not even close. However, dramatic improvement resulted when two carriers were used to match the tenor. Of course, the use of more carriers gave very convincing matches. With the guitar, the decay portion of the tone was fairly easily approximated, but its attack proved somewhat elusive for even fairly high-ordered approximations. This was due to the fact that the guitar tone required a large number (127 in this case) of high-numbered harmonics to adequately represent its initial attack transient. Still, syntheses from matches using a low number of carriers clearly sound guitar-like.

Figures 7 through 12 illustrate amplitude-versus-time plots for the second and fourth harmonics of the trumpet, tenor, and guitar. Amplitude envelopes for the original tones are displayed along with one, three, and five carrier approximations to the original.

With the trumpet, the envelopes of the higher-order matches do a better job at capturing the shape of the original. The tenor's amplitude vibrato is its distinguishing characteristic. The three- and five-carrier matches get the right level on the second harmonic, but only the higher-order match is able to follow the wide amplitude fluctuations in the prominent fourth harmonic properly. Similarly, the three- and five-carrier matches of the guitar do a good job with the second harmonic, but only the five-carrier match tracks the fourth harmonic. In fact, for this particular harmonic, the single-carrier approximation does a better job of matching the peak amplitude and subsequent decay than the three-carrier approximation does. Of course, the single-carrier approximation's performance on most other harmonics, such as the second, is not as good.

Figure 13 illustrates the fitness error (defined by Eq. 5d), plotted against the number of carriers used to

Fig. 7. Second harmonic of trumpet in original and various matches.

Fig. 8. Fourth harmonic of trumpet in original and various matches.

Fig. 9. Second harmonic of tenor in original and various matches.

Fig. 10. Fourth harmonic of tenor in original and various matches.

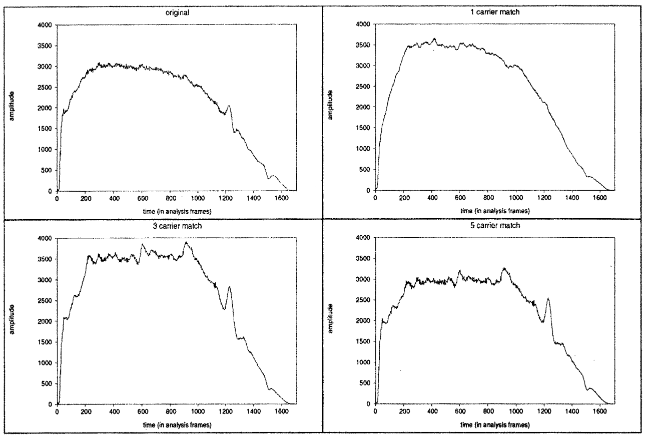


Fig. 7.

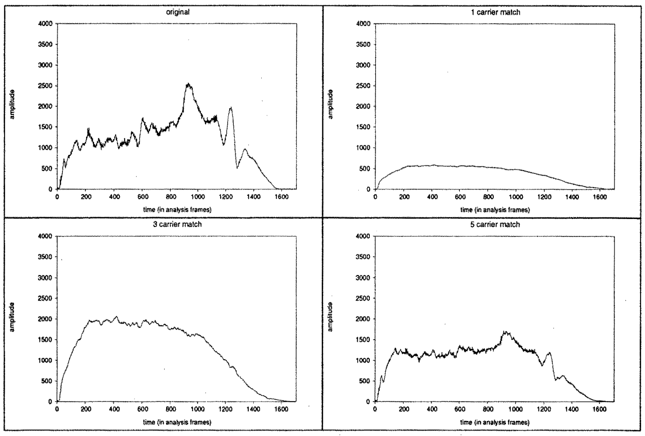


Fig. 8.

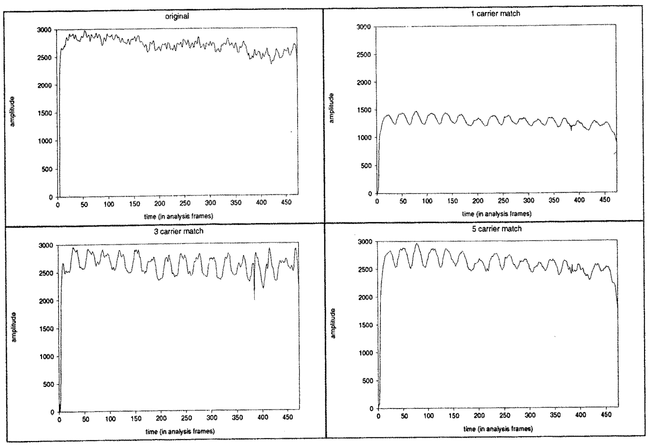


Fig. 9.

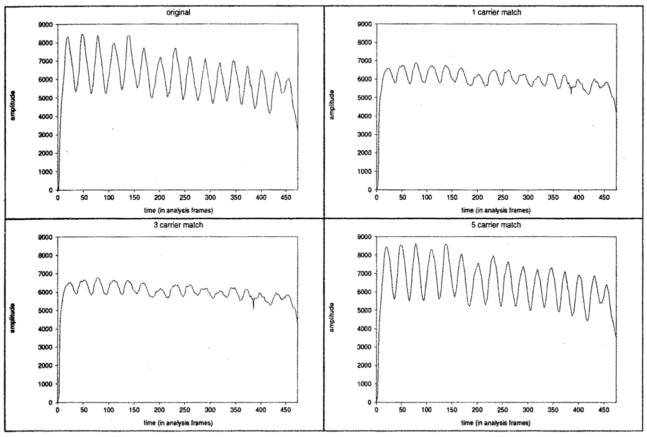


Fig. 10.

match various tones. Here, error is equivalent to the value returned by the least-squares match averaged over all the discrete-time frames. The graph shows that as the number of carriers approaches the number of harmonics, the error does indeed tend to zero. These curves should not be used to compare the relative quality of matches of different sounds, however. For instance, the four-carrier tenor match sounds more like the original than the four-carrier trumpet match despite having twice the error.

The fitness error is thus not an absolute measure of subjective fitness quality, but it is usually indicative of how matches to a particular sound compare with one another. Even for the same sound, there is no guarantee that the result of minimizing the least-squares error will give the best psychoacoustic match. For a small number of carriers (up to three), the best-sounding match may have a slightly greater error than a match which sounds worse. However, a significantly better fitness measure is almost always indicative of a better-sounding match. One way to handle this problem for a small number of carriers is to make several GA runs with different random seeds, then select the best sounding match from among the runs. Because runs with a small number of carriers can be done very quickly, this is an efficient method.

The complete results for a three-carrier match to the trumpet sound are shown in Fig. 14. It shows a plot of the amplitude envelopes $w_j(t)$, the carrier-to-modulator ratios, n_j, and the modulation indices, I_j for this match. We can get a feel for what is happen-

Fig. 11. Second harmonic of guitar in original and various matches.

Fig. 12. Fourth harmonic of guitar in original and various matches.

Fig. 13. Convergence of error with increasing numbers of carriers.

Fig. 14. Individual amplitude envelopes and parameters of three-carrier match to trumpet.

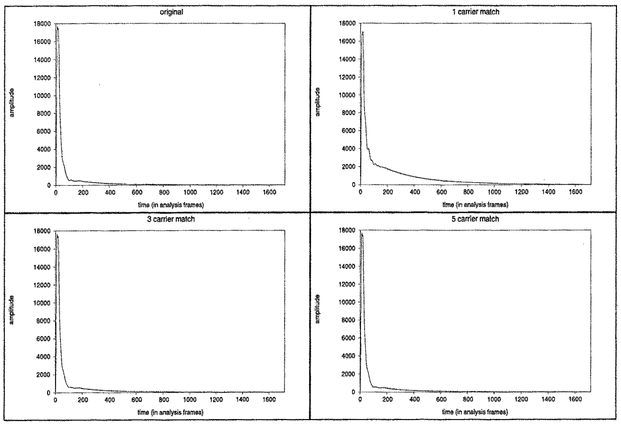


Fig. 11.

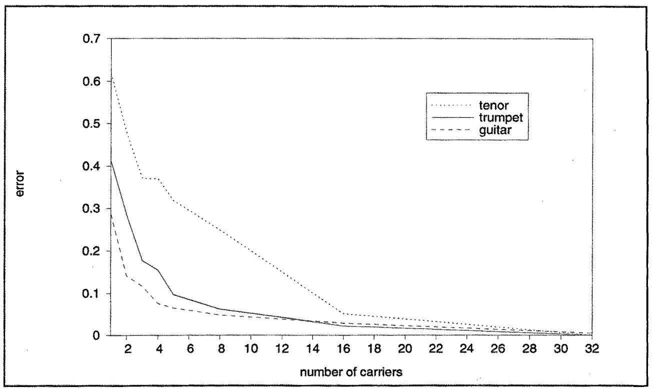


Fig. 13.

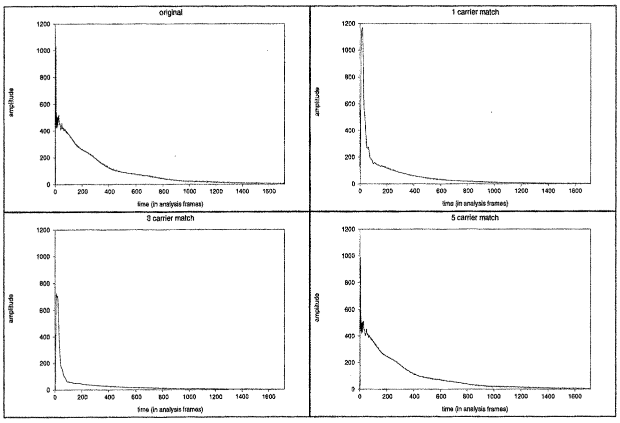


Fig. 12.

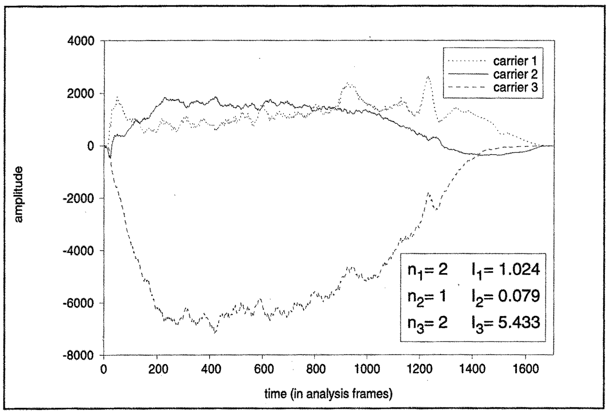


Fig. 14.

ing in the match by examining the parameters. Note that the shape of the third carrier's envelope is nearly identical to that of the original tone shown in Fig. 7, except that it is inverted and scaled by a factor of about two. Because carrier 3 is the only one with a relatively high index, it is definitely the widest-band component of the three. The other two carriers seem to compensate (mostly) for the overaggressiveness of this carrier. Carrier 2 has a near-zero index value (i.e., it is nearly a sine wave) and is centered on the fundamental frequency; therefore it makes a nearly exact match of the original fundamental envelope. Similarly, carrier 1, which has a relatively low index value, primarily controls the amplitude of the second harmonic. However, as we can see in Fig. 7, the three-carrier match's solution for the second harmonic envelope is not perfect. This is not surprising since, in addition to harmonic 2, carrier 1 must contribute intelligently to the match of a few harmonics on either side of it. Meanwhile, partials such as the fourth harmonic are more roughly approximated, as Fig. 8 illustrates. However, we suspect that the ear is more sensitive to details in the lower harmonic envelopes than in the higher ones.

This example is suggestive of how carriers interact in the final mix for any number of carriers. The complexities of this balancing act increase as we add more carriers, which is why FM matching is such a difficult problem. Clearly, the power of a genetic algorithm is needed in searching the space of FM parameters. The parameter landscape is filled with local optima that would trap simple hill-climbing algorithms.

Implementation

The FM matching technique has been implemented on the NeXT workstation. A separate program does the short-time spectrum analysis and writes an analysis file as a result. The matching algorithm begins by reading in the analysis file and, on completion, writes a matching parameter file, an analysis file containing the predicted FM spectra, and a sound file generated by matching FM synthesis. The matching procedure is relatively efficient, running in a few seconds with 1 or 2 carriers, and in a couple of minutes for 5 carriers. Matches with 30 carriers take less than an hour.

Extensions and Generalizations

As an alternative to ordinary FM synthesis with fixed indices, predictive FM (using Bessel functions as in Eq. 2) can be used to fill wavetables with one period of each carrier-modulator pair's waveform. Additive wavetable synthesis, where the output of each wavetable is weighted by a time-varying amplitude envelope and summed to the final output, can then be used to reconstruct the tone. There is no penalty for using this method. Indeed, it is actually more efficient than FM because no modulators are needed.

There is nothing "magic" about using a single modulator and a group of carriers. Other configurations can be adopted with only minor modifications to the scheme. For example, the simplest change involves allowing each carrier to have its own modulator whose frequency is not necessarily the fundamental. In addition, using modulators of modulators presents no overt problems, since the determination of the FM spectra is only a bit more involved (LeBrun 1977). However, in this case the matching optimization subproblems become more difficult to solve because of the increased complexity of interaction between the FM parameters. Preliminary experiments have confirmed this and converged to degraded matches, so more work needs to be done to make this a viable approach.

In this article we have discussed the matching of only single isolated tones. In a practical implementation, we would be synthesizing groups of tones belonging to a particular instrument class over ranges of pitch and dynamics for that instrument. Our current idea for a strategy to handle this case is to use the genetic algorithm to generate basis FM parameters (n_j, I_j), which are optimum over a large collection of spectra obtained from a diverse set of tones produced by the same instrument. Then, for each pitch/dynamic combination, we would determine the set of carrier weighting functions ($w_j(t)$) to best match that case.

Summary and Conclusions

We have introduced a GA-based technique for matching and resynthesizing sounds via FM synthesis. The technique entails analyzing, matching, and finally resynthesizing the tone under consideration. The decomposition of the matching process into tractable subproblems is central to the success of the overall process. For the cases we have tried, the quality of the resulting syntheses indicates that this is a cost-effective approach when a reasonable number of carriers are used. Finally, several extensions and generalizations of the genetic matching technique have been outlined and discussed, establishing the broad applicability of the method. Genetic algorithms provide a powerful and intuitive mechanism to drive the matching process effectively and efficiently . This is in marked contrast to past attempts in FM parameter matching, which have lacked generality and robustness.

With this framework in place, matching can facilitate applications such as data reduction, data stretching, and synthesis by rule. For example, the data reduction of FM matching compared with additive synthesis is proportional to the number of harmonics divided by the number of carriers used in the matching since the amplitude envelopes dominate the storage complexity.

Acknowledgments

This material is based on work supported by the CERL Sound Group at the University of Illinois. Special thanks to members of the CERL Sound Group whose input and feedback has been invaluable in this work. These include Richard Baraniuk, Paul Christensen, Kelly Fitz, Kurt Hebel, Bryan Holloway, Carla Scaletti, and Bill Walker. Thanks also to Lydia

Ayers, Chris Gennaula, Camille Goudeseune, Michael Hamman, and Chris Kriese of the Computer Music Project at the School of Music for their input. This work has also benefitted from conversations with David Goldberg of the Illinois Genetic Algorithm Laboratory (IlliGAL).

References

Beauchamp, J. 1982. "Synthesis by Amplitude and 'Brightness' Matching of Analyzed Musical Instrument Tones." *Journal of the Audio Engineering Society* 30(6): 396–406.

Berger, K. 1963. "Some Factors in the Recognition of Timbre." *Journal of the Acoustical Society of America* 41: 793–806.

Chowning, J. 1973. "The Synthesis of Complex Audio Spectra by Means of Frequency Modulation." *Journal of the Audio Engineering Society* 21(7): 526–534. Reprinted in 1977 in *Computer Music Journal* 1(2): 46–54, and in C. Roads and J. Strawn, eds. 1985. *Foundations of Computer Music*. Cambridge, Massachusetts: MIT Press.

Chowning, J. 1980, "Computer Synthesis of the Singing Voice." In J. Sondberg, ed. *Sound Generation in Wind, Strings, Computers*. Stockholm: Royal Swedish Academy of Music.

Chu, C. 1989. "A Genetic Algorithm Approach to the Configuration of Stack Filters." In *Proceedings of the Third International Conference on Genetic Algorithms and Their Applications*, pp. 112–120.

Clark, M., et al. 1963. "Preliminary Experiments on the Aural Significance of Parts of Tones of Orchestral Instruments and on Choral Tones." *Journal of the Audio Engineering Society* 11(1): 45–54.

Delprat, N., P. Guillemain, and R. Kronland-Martinet. 1990. "Parameter Estimation for Non-linear Resynthesis Methods with the Help of a Time-frequency Analysis of Natural Sounds." In *Proceedings of the 1990 International Computer Music Conference*. San Francisco: International Computer Music Association, pp. 88–90.

Goldberg, D. 1989. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Reading, Massachusetts: Addison-Wesley.

Grey, J., and J. A. Moorer. 1977. "Perceptual Evaluations of Synthesized Musical Instrument Tones." *Journal of the Acoustical Society of America* 62(2): 454–462.

Holland, J. H. 1975. *Adaptation in Natural and Artificial Systems*. Ann Arbor: University of Michigan Press.

Horner, A., and D. Goldberg. 1991. "Genetic Algorithms and Computer-Assisted Music Composition." In *Proceedings of the 1991 International Computer Music Conference*. San Francisco: International Computer Music Association, pp. 479–482.

Justice, J. 1979. "Analytic Signal Processing in Music Computation." *IEEE Transactions on Acoustics, Speech, and Signal Processing* 27(6): 670–684.

LeBrun, M. 1977. "A Derivation of the Spectrum of FM with a Complex Modulating Wave." *Computer Music Journal* 1(4): 51–52.

Maher, R. 1989, "An Approach for the Separation of Voices in Composite Music Signals." Ph.D. diss., University of Illinois at Urbana-Champaign, pp. 19–28, 49–55 .

McAulay, R., and T. Quatieri. 1986. "Speech Analysis/Synthesis Based on a Sinusoidal Representation." *IEEE Transactions on Acoustics, Speech, and, Signal Processing* 34(4): 744–754.

Morrill, D. 1977. "Trumpet Algorithms for Computer Composition." *Computer Music Journal* 1(1): 46–52.

Payne, R. 1987. "A Microcomputer Based Analysis/Resynthesis Scheme for Processing Sampled Sounds Using FM." In *Proceedings of the 1987 International Computer Music Conference*. San Francisco: International Computer Music Association, pp. 282–289.

Press, W., B. Flannery, S. Teukolsky, and W. Vetterling. 1989. *Numerical Recipes*. Cambridge, UK: Cambridge University Press.