Xavier Serra and Julius Smith III
Center for Computer Research in Music and Acoustics
Department of Music, Stanford University
Stanford, California 94305 USA
ucbvax!pixar!ymt!xavier julius@next.com

Spectral Modeling Synthesis: A Sound Analysis/ Synthesis System Based on a Deterministic plus Stochastic Decomposition

When generating musical sound on a digital computer, it is important to have a good model whose parameters provide a rich source of meaningful sound transformations. Three basic model types are used widely today for musical sound generation: instrument models, spectrum models, and abstract models. Instrument models attempt to parameterize a sound at its source, such as a violin, clarinet, or vocal tract. Spectrum models attempt to parameterize a sound at the basilar membrane of the ear, discarding whatever information the ear seems to discard in the spectrum. Abstract models, such as FM, attempt to provide musically useful parameters in an abstract formula.

This paper addresses the second category of synthesis technique: spectrum modeling. It describes a technique called *spectral modeling synthesis* (SMS), that models time-varying spectra as (1) a collection of sinusoids controlled through time by piecewise linear amplitude and frequency envelopes (the *deterministic* part), and (2) a time-varying filtered noise component (the *stochastic* part). The analysis procedure first extracts the sinusoidal trajectories by tracking peaks in a sequence of short-time Fourier transforms. These peaks are then removed by spectral subtraction. The remaining "noise floor" is then modeled as white noise through a time-varying filter. A piecewise linear approximation to the upper spectral envelope of the noise is computed for each successive spectrum, and the stochastic part is synthesized by means of the overlap-add technique. The SMS technique has proved to give general, high quality transformations for a wide variety of musical signals.

Computer Music Journal, Vol. 14, No. 4, Winter 1990,
© 1990 Massachusetts Institute of Technology.

Background

Additive synthesis is the original spectrum modeling technique. It is rooted in Fourier's theorem, which states that any periodic waveform can be modeled as a sum of sinusoids at various amplitudes and harmonic frequencies.

Additive synthesis was among the first synthesis techniques in computer music. In fact, it was described extensively in the first article of the first issue of the *Computer Music Journal* (Moorer 1977).

In the early 1970s, Moorer developed a series of analysis programs to support additive synthesis. He first used the heterodyne filter to measure the instantaneous amplitude and frequency of individual sinusoids (Moorer 1973). The *heterodyne filter* implements a single frequency bin of the discrete Fourier transform (DFT), using the rectangular window. The magnitude and phase derivative of the complex numbers produced by the sliding DFT bin provided instantaneous amplitude and frequency estimates. The next implementation (Moorer 1978) was based on the digital phase vocoder (Portnoff 1976). In this

system, the fast Fourier transform (FFT) was used to provide, effectively, a heterodyne filter at each harmonic of the fundamental frequency. The use of a nonrectangular window gave better isolation among the spectral components.

For years, Moorer's phase vocoder provided analysis support for additive synthesis at CCRMA. In the course of his thesis work, Grey (1975) demonstrated that natural instrument sounds, such as oboe and clarinet, could be successfully modeled as a sum of sinusoidal oscillators with piecewise linear amplitude envelopes. The degree to which the amplitude envelopes could be simplified by piecewise-linear approximation was surprising; it was not uncommon for a 100:1 data reduction to sound as good as the original.

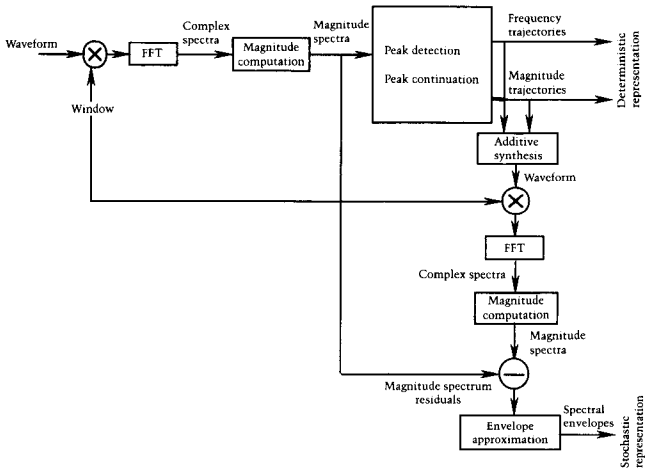
The main problem with the phase vocoder was that inharmonic sounds with deep vibrato were difficult to analyze. It is well known that the FFT can be regarded as a fixed filter bank or graphic equalizer. If the length of the FFT is N, then there are N narrow bandpass filters, slightly overlapping, equally spaced between 0 Hz and the sampling rate. In the phase vocoder, the instantaneous amplitude and frequency are computed only for each channel filter or bin. A consequence of using a fixed-frequency filter bank is that the frequency of each sinusoid is not normally allowed to vary outside the bandwidth of its channel, unless one is willing to combine channels in some fashion that requires extra work. (The channel bandwidth is nominally the sampling rate divided by the FFT length.) Also, the analysis system was really set up for harmonic signals—you could analyze a piano if you had to, but the progressive sharpening of the partials meant that there would be frequencies where a sinusoid would be in the crack between two adjacent FFT bins. This was not an insurmountable condition (the adjacent bins could be combined intelligently to provide accurate amplitude and frequency envelopes), but it was inconvenient and outside the original scope of the analysis framework of the phase vocoder.

The PARSHL program was developed at CCRMA for the purpose of supporting inharmonic and pitch-changing sounds (Smith and Serra 1987). PARSHL was a simple application of FFT peak-tracking technology commonly used in the Navy signal processing community (General Electric 1977; Wolcin 1980a; 1980b; Smith and Friedlander 1984). As in the phase vocoder, a series of FFT frames is computed by PARSHL. However, instead of writing out the magnitude and phase derivative of each bin, the FFT is searched for peaks, and the largest peaks are tracked from frame to frame. The principal difference in the analysis is the replacement of the phase derivative in each FFT bin by interpolated magnitude peaks across FFT bins. This approach is better suited for analysis of inharmonic sounds.

At about the same time, Quatieri and McAulay developed independently a facility similar to PARSHL for analyzing speech (McAulay and Quatieri 1984; 1986). Their system differed in the following ways: (1) peaks were found as changes in slope of the magnitude spectrum, whereas PARSHL finds the maximum magnitude over all frequencies and "pulls out" the whole "hill"; (2) spectral peaks were not interpolated in frequency or amplitude as they are in PARSHL; (3) amplitude and frequency envelopes were not simplified to small numbers of piecewise linear breakpoints (a breakpoint was retained for each frame); (4) peak association across frames (including "birth" and "death" criteria for the line tracks) was solved by a different algorithm; and (5) PARSHL supports additional constraints, such as limiting the peak search to a specific frequency interval, specifying a maximum glissando slope before dissociating peaks connected across adjacent frames, or rejecting peaks below a minimum dB level and/or width. Both systems were built on top of the short-time Fourier Transform facility (Allen 1977).

The PARSHL program works well for most sounds created by simple physical vibrations or driven periodic oscillations. It goes beyond the phase vocoder to support spectral modeling of inharmonic sounds. A problem with PARSHL, however, is that it is unwieldy to represent noiselike signals, such as the attack of many instrumental sounds. Using sinusoids to simulate noise is extremely expensive because, in principle, noise consists of sinusoids at every frequency within the band limits. This motivated the next round of improvements, which is described in this paper.

Fig. 1. Block diagram of the analysis part of SMS.



The Deterministic plus Stochastic Model

A sound model assumes certain characteristics of the sound waveform or the sound generation mechanism. In general, every analysis/synthesis system has an underlying model. The SMS technique assumes the input sound to be composed of a deterministic plus a stochastic component.

A deterministic signal is traditionally defined as anything that is not noise (i.e., an analytic signal, or perfectly predictable part, predictable from measurements over any continuous interval). However, in the present discussion the class of deterministic signals considered is restricted to sums of quasi-sinusoidal components (sinusoids with piecewise linear amplitude and frequency variation). Each sinusoid models a narrowband component of the original sound and is described by an amplitude and a frequency function.

A stochastic, or noise, signal is fully described by its amplitude probability density versus frequency, or its power spectral density. When a signal is assumed to be stochastic, it is not necessary to preserve either the instantaneous phase or the exact details of individual FFT frames.

Therefore, the input sound $s(t)$ is modeled as the sum of a series of sinusoids plus a noise signal,

$$s(t) = \sum_{r=1}^{R} A_r(t) \cos[\theta_r(t)] + e(t),$$

where $A_r(t)$ and $\theta_r(t)$ are the instantaneous amplitude and phase of the r^{th} sinusoid, and $e(t)$ is the noise component at time t (in seconds).

The model assumes that the sinusoids are stable partials of the sound and that each one has a slowly changing amplitude and frequency. The instantaneous phase is then taken to be the integral of the instantaneous frequency $\omega_r(t)$, and therefore satisfies

$$\theta_r(t) = \int_0^t \omega_r(\tau)d\tau + \theta_r(0),$$

where $\omega(t)$ is the frequency in radians per second and r is the sinusoid number.

By assuming that $e(t)$ is a stochastic signal, it can be described as filtered white noise,

$$e(t) = \int_0^t h(t,\tau)u(\tau)d\tau,$$

where $u(\tau)$ is white noise and $h(t,\tau)$ is the impulse response of a time-varying filter at time t. That is, the residual is modeled by the convolution of white noise with a time-varying frequency-shaping filter.

Description of the SMS Technique

Figures 1 and 2 show the block diagrams for the analysis and synthesis parts of the SMS technique. The first step is the derivation of a series of magnitude spectra of the input sound by computing the FFT of every windowed portion of the waveform: i.e., computation of the sort-time Fourier transform (STFT). From the series of magnitude spectra the prominent peaks are detected in each spectrum. These peaks are then organized into frequency trajectories by means of a peak continuation algorithm. The relevance of this algorithm is that it extracts the stable sinusoids present in the original sound (the deterministic component).

The stochastic part of the waveform is calculated by first computing the STFT of the deterministic component, in the same way that the STFT of the original waveform was obtained, and then subtracting each magnitude spectrum from the corresponding spectrum of the original waveform. The envelope of each "residual" spectrum is then derived by

Fig. 2. Block diagram of the synthesis part of SMS.

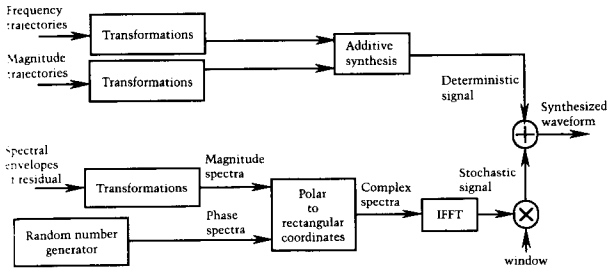
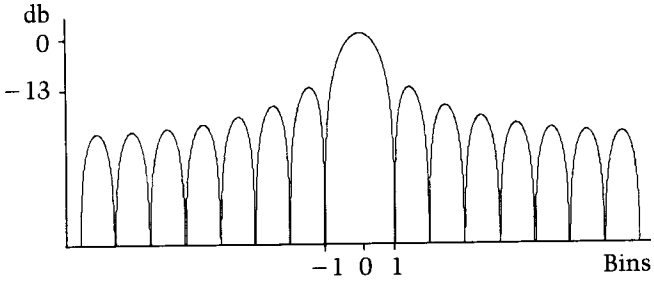


Fig. 3. Magnitude spectrum of a rectangular window.



performing a line-segment approximation. These envelopes represent the stochastic signal.

The deterministic signal (i.e., the sinusoidal component) results from the magnitude and frequency trajectories, or their transformation, by generating a sine wave for each trajectory (i.e., additive synthesis).

The stochastic signal is the result of creating a complex spectrum (i.e., magnitude and phase spectra) for every spectral envelope of the residual, or its modification, and performing an inverse-STFT (using the overlap-add method to form the final output). The magnitude spectrum is the envelope itself, and the phase spectrum is generated by a random number generator. This process corresponds to the filtering of white noise by a filter with a frequency response equal to the spectral envelope. The following sections describe each step of the system.

Computation of the Magnitude Spectra

The analysis part of SMS starts by computing a set of magnitude spectra using the STFT (Allen 1977; Allen and Rabiner 1977). This transform can be understood as a time-varying DFT. It is defined as

$$X_l(k) \triangleq \sum_{n=0}^{N-1} w(n)x(n + lH)e^{-j\omega_k n}, \quad l = 0, 1, \ldots$$

where $w(n)$ is a real window that determines the portion of the input signal $x(n)$ that receives emphasis at a particular frame l. H is the hop size, or time advance, of the window. Therefore, the STFT computes a DFT (i.e., a spectrum) at every frame l, advancing with a hop size of H so as to slide the window $w(n)$ along the sequence $x(n)$.

The output of the STFT is a series of spectra, one for every frame l of the input waveform. Each spectrum $X_l(k)$ is a complex valued function of bin-number k, converted to a magnitude function by

$$A_l(k) = |X_l(k)| = \sqrt{a^2(k) + b^2(k)},$$

where $|X_l(k)|$ is the magnitude spectrum, and $a(k)$ and $b(k)$ are the real and imaginary parts of the complex value returned by the DFT for bin k.

Analysis Window

The choice of the analysis window is important. It determines the trade-off of time versus frequency resolution, which affects the smoothness of the spectrum and the detectability of different sinusoidal components. The most commonly used windows are called Rectangular, Hamming, Hanning, Kaiser, Blackman, and Blackman-Harris. Harris (1978) gives a good discussion of these and many other windows.

All the standard windows are real and symmetric and have a frequency spectrum with a sinclike $(\sin(x)/x)$ shape (Fig. 3). For the purposes of SMS, and in general for any sound analysis/synthesis application, the choice of window type is mainly determined by two of its spectral characteristics: (1) the width of the main lobe, defined for present purposes as the number of bins between zero crossings on either side of the main lobe when the DFT length equals the window length; and (2) the highest side-lobe level, which measures how many dB down the highest side lobe is from the main lobe. Ideally, we want a narrow main lobe (i.e., good frequency resolution) and a very low side-lobe level

(i.e., no cross-talk between DFT channels). The choice of window determines this trade-off. For example, the rectangular window has the narrowest main lobe, two bins, but the first side lobe is very high—13 dB relative to the main-lobe peak. The Hamming window has a wider main lobe, four bins, and the highest side lobe is 43 dB down. A very different window, the Kaiser, allows control of the trade-off between the main-lobe width and the highest side-lobe level. If a narrower main-lobe width is desired, then the side-lobe level will be higher, and vice versa. Since control of this trade-off is valuable, the Kaiser window is a good general purpose choice. (See Serra [1989] for a more extensive discussion.)

The window length must be sufficient to resolve the most closely spaced sinusoidal frequencies. A nominal choice for periodic signals is about four periods.

Computation of the DFT

Once a section of the waveform has been windowed, the next step is to compute the spectrum using the DFT. For practical purposes, the FFT should be used whenever possible, but this requires the length of the analyzed signal to be a power of two. This can be accomplished by taking any desired window length and "zero padding," i.e., filling with zeros out to the length required by the FFT. This not only allows use of the FFT algorithm, but computes a smoother spectrum as well. Zero-padding in the time domain corresponds to interpolation in the frequency domain.

The size of the FFT, N, is normally chosen to be the first power of two that is at least twice the window length M, with the difference $N - M$ filled with zeros. If B_s is the number of samples in the main lobe when the zero-padding factor is 1 ($N = M$), then a zero-padding factor of N/M gives B_sN/M samples for the same main lobe (and same main-lobe bandwidth). The zero-padding (interpolation) factor N/M should be large enough to enable an accurate estimation of the true maximum of the main lobe. That is, since the window length is not an exact number of periods for every sinusoidal frequency, the spectral peaks do not, in general, occur at FFT bin frequencies (multiples of f_s/N). Therefore, the bins must be interpolated to estimate peak frequencies. Zero padding is one type of spectral interpolation.

Choice of Hop Size

Once the spectrum has been computed at a particular frame in the waveform, the STFT hops along the waveform and computes the spectrum of the next section in the sound. This hop size H (i.e., how much the analysis time origin is advanced from frame to frame) is an important parameter. Its choice depends very much on the purpose of the analysis. In general, more overlap will give more analysis points and therefore smoother results across time, but the computational expense is proportionally greater. A general and valid criterion is that the successive frames should overlap in time, in such a way that all the data are weighted equally. A good choice is the window length divided by the main lobe width in bins (Allen 1977). For example, a practical value for the Hamming window is to use a hop size equal to one-fourth of the window size.

Peak Detection

Once the set of spectra of a sound is computed, the system extracts the prominent peaks of each spectrum. A peak is defined as a local maximum in the magnitude spectrum. However, not all the peaks are equally prominent in the spectrum, and it is important to have control over their selection. This is done by measuring the height of each peak in relation to its neighboring valleys, where the neighboring valleys are the closest local minima on both sides of the peak. Also, not all the peaks of the same height are equally relevant perceptually; their amplitude and frequency are important. Thus, it is useful to specify frequency and magnitude ranges where the search for peaks takes place.

Due to the sampled nature of the spectra returned by the STFT, each peak is accurate only to

Fig. 4. Peak detection on a spectrum of a piano attack sound.

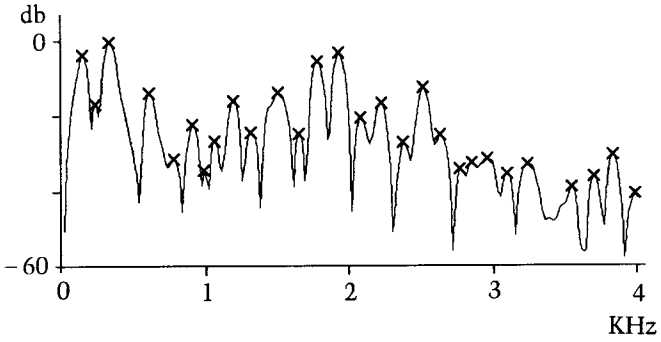
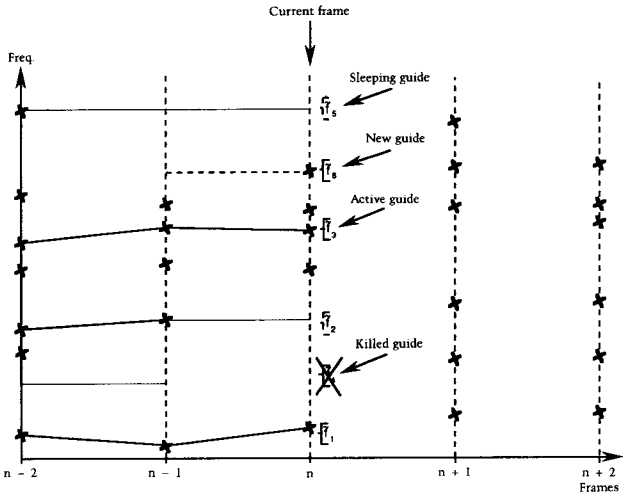


Fig. 5. Illustration of the peak continuation algorithm. The frequency-guides *have just been continued through frame* n.



within half a bin. A bin represents a frequency interval of f_s/N Hz, where N is the FFT size, and f_s is the sampling rate. As we saw in the previous section, zero padding in the time domain increases the number of DFT bins per Hz and thus increases the accuracy of simple peak detection. However, to obtain good frequency accuracy, the zero-padding factor required is very large. A more efficient spectral interpolation scheme is to zero pad only enough so that parabolic spectral interpolation, using only three bins immediately surrounding the maximum-magnitude bin, can be used to refine the accuracy of the estimate (Serra 1989). Note that parabolic interpolation would be exact in the case of log-magnitude spectra using a Gaussian window. Figure 4 shows the result of the peak detection on a spectrum.

Peak Continuation

Once the spectral peaks have been detected, a subset of them is organized by the peak continuation algorithm into peak trajectories, where each trajectory represents a stable sinusoid. The design of such an algorithm can be approached as a line detection problem, where out of a surface of discrete points, each one being a peak, the algorithm finds lines according to the characteristics imposed by the model.

The algorithm is intended to work for a variety of sounds. The behavior of a partial, and therefore the way to track it, varies depending on the sound. Whether the sound source is speech, a violin, a gong, an animal, etc., the time evolution of the component partials will vary. Thus, apart from being general, the algorithm requires some knowledge of the characteristics of the sound that is being analyzed. In the current algorithm there is no attempt to make the process completely automatic. The user is expected to know some of the characteristics of the sound beforehand, specifying them through a set of parameters.

The basic idea of the peak continuation algorithm is that a set of *frequency guides* advances in time through the spectral peaks, looking for the appropriate ones (according to the specified constraints), and forming trajectories out of them. The instantaneous state of the guides, their frequency, is kept in the variables $\tilde{f}_1, \tilde{f}_2, \tilde{f}_3, \ldots, \tilde{f}_p$, where p is the number of existing guides. These values are continuously updated as the guides are turned on, advanced, and finally turned off.

To describe the peak continuation algorithm, let us assume that the frequency guides are currently at frame n. Suppose that the guide frequencies at the current frame are $\tilde{f}_1, \tilde{f}_2, \tilde{f}_3, \ldots, \tilde{f}_p$, where p is the number of existing guides. We want to continue the p guides through the peaks of frame n with frequencies $g_1, g_2, g_3, \ldots, g_m$, thus continuing the corresponding trajectories. Figure 5 illustrates the algorithm. There are three steps: guide advancement, update of the guide values, and start of new guides. These steps are described below.

Guide Advancement

Each guide is advanced through frame n by finding the peak closest in frequency to its current value. Guide r claims frequency g_i for which $|\tilde{f}_r - g_i|$ is a minimum. The three possible situations are as follows.

- If a match is found within the maximum deviation allowed, the guide is continued (unless there is a conflict to resolve, as described below). The selected peak is then incorporated into the corresponding trajectory.
- If no match is found, it is assumed that the corresponding trajectory must turn off entering frame n, and its current frequency is matched to itself with 0 magnitude. Since the trajectory amplitudes are ramped linearly from one frame to the next, the terminating trajectory ramps to 0 over the duration of one hop size. Whether or not the actual guide is killed depends on the allowed sleeping time.
- If a guide finds a match that has already been claimed by another guide, we give the peak to the guide that is closest in frequency, and the loser looks for another match. If the current guide loses the conflict, it simply picks the best available nonconflicting peak within the allowed frequency range. If the current guide wins the conflict, it calls the assignment procedure recursively on behalf of the dislodged guide. When the dislodged guide finds the same peak and wants to claim it, it sees there is a conflict that it loses and moves on. This process is repeated for each guide, solving conflicts recursively, until all possible matches are made.

Update of the Guide Values

Once all the existing guides and their trajectories have been continued through frame n, the guide frequencies are updated. There are two possible situations.

- If a guide finds a continuation peak, its frequency is updated from $\tilde{f}_r$ to $\tilde{h}_r$ according to

$$\tilde{h}_r = \alpha(g_i - \tilde{f}_r) + \tilde{f}_r, \quad \alpha \in [0,1]$$

  where g_i is the frequency of the peak that the guide has found at frame n, and α is the given contribution that the peak makes to the guide. When $\alpha = 1$, the frequency of the peak trajectory is the same as the frequency of the guide, so the difference between guide and trajectory is lost.
- If a guide has not found a continuation peak for the allowed sleeping time, the guide is killed at frame n. If it is still under the sleeping time, it keeps whatever value it already had. In order to distinguish between guides that find a continuation peak from the ones that do not but still are alive, we refer to the first ones as *active guides* and the second ones as *sleeping guides.*

Start of New Guides

New guides, and therefore new trajectories, are created from the peaks of frame n that are not incorporated into trajectories by the existing guides.

A guide is created at frame n by searching through the unclaimed peaks of the frame for the one with the highest magnitude that is separated from every existing guide by a given minimum frequency separation. The frequency value of the selected peak is the frequency of the new guide. The actual trajectory is started in the previous frame, $n - 1$, where its amplitude value is set to 0 and its frequency value to the same as the current frequency, thus ramping in amplitude to the current frame. This process is recursively done until there are no more unclaimed peaks in the current frame, or the number of guides has reached the maximum allowed.

The attack portion of most sounds is quite noisy, and the search for partials is harder in such a rich spectrum. A useful modification to the algorithm is to start the process from the end of the sound: that is, to start tracking the peaks from the last frame and work towards the front. The tracking process encounters the end of the sound first, and since this is a very stable part in most instrumental sounds,

the algorithm finds a very clear definition of the partials. When the guides arrive at the attack, they are already tracking the main partials and can reject irrelevant peaks appropriately, or at least evaluate them with some knowledge acquired from the rest of the analysis.

Variations on the Algorithm for Harmonic Sounds

When the sound to be analyzed is known to be harmonic, the peak continuation algorithm can be specialized. For this case, the *frequency guides* are circumscribed to track harmonic frequencies. Two major changes result with respect to the general algorithm: there is a specific fundamental frequency at every frame, and each guide tracks a specific harmonic number.

In order to accommodate these changes, the steps involved in this new algorithm are slightly different. At the current frame the succession of steps is now the detection of the fundamental frequency, and guide advancement. The only new step is the detection of the fundamental frequency; the guide advancement is the same as for the general case.

Before advancing the guides through frame n, a pitch detection algorithm searches for the fundamental frequency of frame n. If this is found, the guide values are reset to the harmonic series of the new fundamental, without considering the values of the previous frame. If the fundamental is not found, the guides keep their frequency values.

Given the set of peaks of frame n, with magnitude and frequency values for each one, there are many possible fundamental detection strategies. In the current application, we are dealing with single-source sounds and assuming that a fundamental-frequency peak exists. With these restrictions, a simple algorithm is designed that suffices for this situation. It is based on finding the M highest peaks at frame n and then searching for the peak that is a fundamental for all of them (we are currently using $M = 3$). By choosing the highest peaks, it is assured that they are good harmonic partials, therefore they have to be multiples of a fundamental.

Representation of the Deterministic Part

The output of the peak continuation algorithm is a set of peak trajectories. These represent the deterministic component: i.e., the partials of the analyzed sound. Each peak is a pair of numbers of the form $\{\hat{A}_r(l), \hat{\omega}_r(l)\}$, where $\hat{A}$ and $\hat{\omega}$ are the amplitude and frequency, respectively, for each frame l and each trajectory r. The pairs corresponding to a trajectory r are interpreted as breakpoints for amplitude and frequency functions, one breakpoint for each frame l. From these functions, a series of sinusoids can be synthesized that reproduce the deterministic part of the sound.

These amplitude and frequency functions can be further processed to achieve a data reduction of the representation. A data reduction strategy is to perform a line-segment approximation on each function, thus reducing the number of breakpoints (Grey 1975; Strawn 1980). However, for the purpose of easy manipulation of the representation it is useful to have equally spaced points along each function, and thus it may be better to keep one breakpoint per frame as returned by the analysis, unless data reduction is a priority. Another alternative for data reduction is to combine groups of similar functions into a single one, thus reducing the number of functions.

Deterministic Synthesis

Given the representation of the deterministic part of the sound, SMS generates the time domain waveform with an additive synthesis technique. From the amplitude and frequency functions, $\hat{A}_r(l)$ and $\hat{\omega}_r(l)$, a frame of the deterministic sound is obtained by

$$d^l(m) = \sum_{r=1}^{R^l} \hat{A}_r^l \cos[m\hat{\omega}_r^l], \qquad m = 0, 1, 2, \ldots, H-1$$

where R^l is the number of trajectories present at frame l, and H is the length of the synthesis frame (without any time scaling H is the analysis hop

size). The final sound $d(n)$ results from the juxtaposition of all the synthesis frames. To avoid clicks at the frame boundaries, the parameters $(\hat{A}_r^l, \hat{\omega}_r^l)$ are smoothly interpolated from frame to frame.

The instantaneous amplitude $\hat{A}(m)$ is obtained by linear interpolation,

$$\hat{A}(m) = \hat{A}^{l-1} + \frac{(\hat{A}^l - \hat{A}^{l-1})}{H} m,$$

where $m = 0, 1, \ldots, H - 1$ is the time sample in the l frame.

The instantaneous phase is taken to be the integral of the instantaneous frequency, where the instantaneous radian frequency $\hat{\omega}(m)$ is also obtained by linear interpolation,

$$\hat{\omega}(m) = \hat{\omega}^{l-1} + \frac{(\hat{\omega}^l - \hat{\omega}^{l-1})}{H} m,$$

and the instantaneous phase for the rth trajectory is

$$\hat{\theta}_r(m) = \hat{\theta}_r(l - 1) + \hat{\omega}_r(m)m.$$

Finally, the synthesis equation becomes

$$d^l(m) = \sum_{r=1}^{Rl} \hat{A}_r^l(m) \cos[\hat{\theta}_r^l(m)],$$

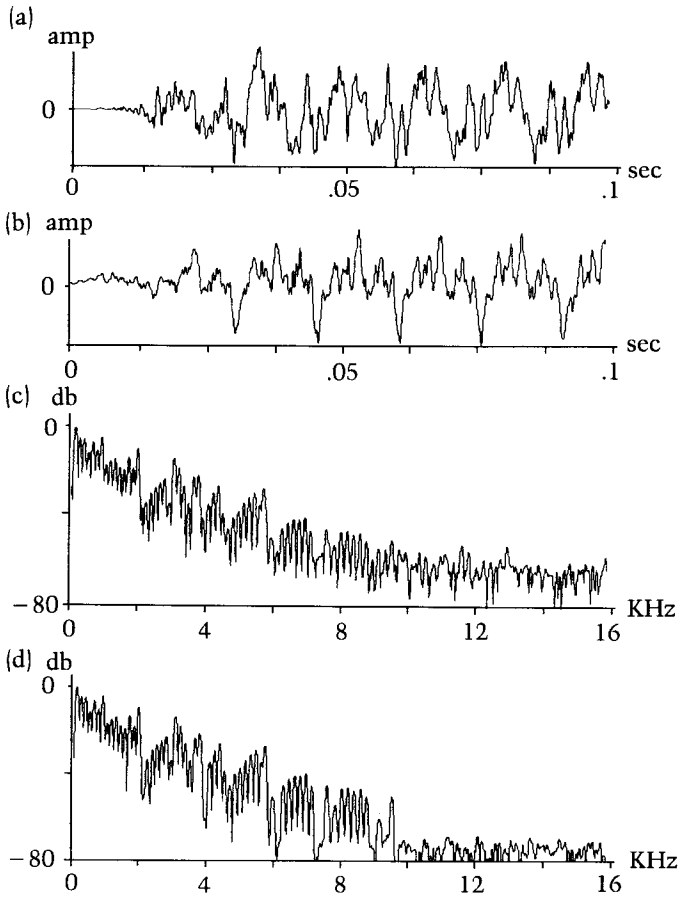
where $\hat{A}(m)$ and $\hat{\theta}(m)$ are the calculated instantaneous amplitude and phase.

Computation of the Stochastic Part

Once the deterministic component of the sound has been detected, the next step is to obtain the residual, which in a simplified form becomes the stochastic component.

Since the deterministic component does not preserve the phases of the original sound, a time domain subtraction cannot be performed. For a method that allows a time domain subtraction see Serra (1989). However, since the magnitude and frequency of each sinusoid are preserved, the magnitude spectrum of both signals are comparable, as shown in Fig. 6. Accordingly it is possible to perform a frequency-domain subtraction from the magnitude spectra of both signals. The result is a set of magnitude spectrum residuals.

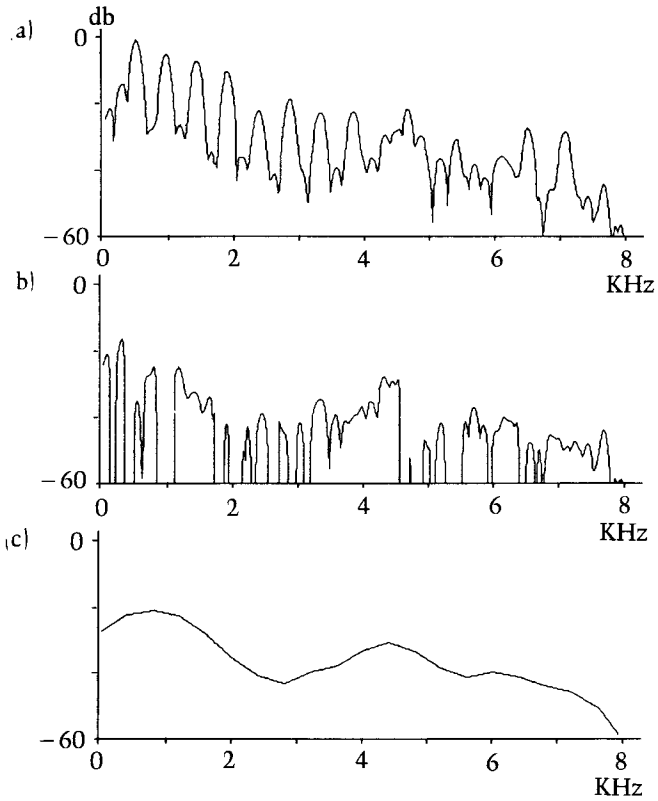
Fig. 6. Example of the deterministic synthesis without phase tracking: waveform from an original piano tone (a), deterministic component without phase tracking (b), magnitude spectrum of original sound (c), magnitude spectrum of deterministic component (d).



One of the underlying assumptions of the current model is that the residual is a stochastic signal. Such an assumption implies that the residual sound is fully described by its amplitude and its general frequency characteristics. It is unnecessary to keep either the instantaneous phase or the exact frequency information. Based on this, the stochastic residual can be completely characterized by the envelopes of the magnitude-spectrum residuals: i.e., these envelopes keep the amplitude and the general shape of the residual spectrum. The set of envelopes forms the stochastic representation.

The computation of the stochastic representation involves the subtraction of each magnitude spectrum of the deterministic component from the corresponding magnitude spectrum of the original sound, and the approximation of each residual spectrum with an envelope. Each step is described below.

Fig. 7. Example of the stochastic computation: magnitude spectrum from a piano tone (a), residual spectrum (b), residual approximation (c).



Computation of the Spectral Residuals

The first step in obtaining the stochastic component is to subtract the set of magnitude spectra of the deterministic signal from that of the original sound. This results in the magnitude-spectrum residuals shown in Fig. 7. For this to be feasible, the spectra to be subtracted have to be comparable and therefore have to be computed in the same manner. The STFTs from which they are obtained must use the same analysis window, window length, FFT size, and hop size.

Given that the magnitude spectrum of the original sound at frame l is $|X_l(k)|$ and that of the deterministic signal is $|D_l(k)|$, then the residual is

$$|E_l(k)| = ||X_l(k)| - |D_l(k)||.$$

Approximation of the Spectral Residual

Assuming that the residual signal is quasi-stochastic, each magnitude-spectrum residual can be approximated by its envelope, since only its shape contributes to the sound characteristics.

This type of problem is generally solved by performing some sort of curve fitting (Strawn 1980): i.e., finding a function that matches the general contour of a given curve, which in our case is a magnitude spectrum. Standard techniques are spline interpolation, the method of least squares, or straight-line approximations. For the purposes of our system, a simple line-segment approximation is accurate enough and gives the desired flexibility.

Another practical alternative is to use a type of least-squares approximation called linear predictive coding, LPC (Markel and Gray 1976). LPC is a popular technique used in speech research for fitting an nth-order polynomial to a magnitude spectrum. It is sufficient to say that the line-segment approach is more flexible than LPC, and even though LPC results in fewer analysis points, the flexibility is considered more important here.

The particular line-segment approximation performed here is done by stepping through the magnitude spectrum and finding local maxima in every section,

$$\tilde{E}_l(q) = \max_k(|E_l(k + qH)|),$$
$$k = -M/2, -M/2 + 1, \ldots, 0, \ldots, M/2 - 2, M/2 - 1$$
$$q = 0, 1, \ldots$$

where H is the hop size, M the window size (or size of the section) and $\tilde{E}_l(q)$ is the maximum of section q at frame l. The resulting points are linearly interpolated to create the spectral envelope (as shown in Fig. 7). The accuracy of the fit is given by the hop size, which is set depending on the sound complexity.

Representation of the Stochastic Part

The stochastic analysis returns an envelope $\tilde{E}_l(q)$ for every frame l, where q is the breakpoint number in the envelope, $q = 0, 1, \ldots, Q - 1$. These envelopes can be interpreted differently depending on which variable, l or q, is considered fixed. When l is fixed, the interpretation is as a frequency-shaping

filter for frame l. When q is fixed, the interpretation is, as an amplitude-modulated bandpass filter, centered at $f_s q/2Q$ Hz and with a bandwidth of $f_s/2Q$ Hz.

These frequency envelopes or time functions (depending on the interpretation) can be simplified and smoothed as in the deterministic representation. As with that representation, it is also useful to keep the same number of breakpoints on both the frequency and the time axes.

Stochastic Synthesis

The synthesis of the stochastic component can be understood as the generation of a noise signal that has the frequency and amplitude characteristics described by the spectral envelopes of the stochastic representation. The intuitive operation is filtering white noise with these frequency envelopes: that is, performing a time-varying filtering of white noise. In practice, the SMS technique generates the stochastic signal by an overlap-add synthesis technique from the spectral envelopes. The inverse Fourier transform of each envelope is computed and the resulting waveforms are overlapped and added.

Before the inverse-STFT is performed, a complex spectrum (i.e., magnitude and phase spectra) is obtained from each frequency envelope. The magnitude spectrum is generated by interpolating the approximation $\tilde{E}_l(q)$ of length Q to a curve of length $N/2$, where N is the FFT size. The FFT size is the first power of 2 that is bigger than the synthesis window (discussed below) plus Q. There is no phase information in the stochastic representation, but since the phase spectrum of noise is a random signal, the phase spectrum can be created with a random number generator. To avoid a periodicity at the frame rate, different values are generated at every frame. Therefore the magnitude and phase spectra at frame l are

$$A_l(k) = \tilde{E}'_l(k),$$

$$\Theta_l(k) = \pi - \text{ran}(2\pi),$$

where $\tilde{E}'_l(k)$ is the interpolated spectral envelope and ran(2π) is a function that produces random numbers in the range from $0-2\pi$.

From the interpolated magnitude envelope and the random phase spectrum, the complex spectrum $\hat{E}_l(k)$ results from a change of coordinates,

$$\text{Re}\{\hat{E}_l(k)\} = A_l(k)\cos[\Theta_l(k)],$$

$$\text{Im}\{\hat{E}_l(k)\} = A_l(k)\sin[\Theta_l(k)].$$

Its inverse Fourier transform gives one frame of the noise waveform,

$$\hat{e}'_l(m) = \frac{1}{N}\sum_{k=-N/2}^{N/2-1} \hat{E}_l(k)e^{j\omega_k m}, \quad m = 0, 1, \ldots, N-1$$

The waveform $\hat{e}'_l(m)$ is a constant-amplitude waveform of size N, where N is the FFT size. Since the phase spectrum used is not the result of an analysis process (with windowing of a waveform, zero padding, and FFT computation), the resulting signal does not taper to 0 at the boundaries. This is because a phase spectrum with random values corresponds to a phase spectrum of a rectangular-windowed noise waveform of size N. In order to succeed in the overlap-add resynthesis (i.e., to obtain smooth transitions between frames) we need a smoothly windowed waveform of size M, where M is the synthesis-window length. This window length depends on the window used, but a practical choice is to use a length four times the hop size. Note that this length is independent of the analysis-window length, and it is only related to the analysis hop size (in the case where no time scaling is performed). Therefore the resulting waveform $\hat{e}'_l$f(m) is multiplied by a length M window,

$$\hat{e}_l(m) = \hat{e}'_l(m)w(m). \quad m = 0, 1, \ldots, M-1$$

There is no reason to use the same window as in the STFT analysis, nor to use a very sophisticated one. A simple Hanning window suffices. Then the stochastic signal results from the overlap and add of these windowed waveforms,

$$\hat{e}(n) = \sum_{l=0}^{L-1} \hat{e}_l(n - lH),$$

where H is the analysis hop size and l is the frame number.

Representation of Timbral Modifications

The deterministic analysis results in a set of amplitude and frequency functions, $\hat{A}_r(l)$ and $\hat{\omega}_r(l)$, where r is the function number and l the breakpoint number in each function. The stochastic analysis results in a set of spectral envelopes, $\tilde{E}_l(q)$, where q is the breakpoint number in the envelope. From these representations, a great number of sound transformations are possible.

Time-scale modifications are accomplished in both representations by resampling the analysis points in time. This is done by changing the synthesis frame size (or the hop size in the case of the stochastic synthesis) and results in slowing down or speeding up the sound while maintaining pitch and formant structure. A time-varying frame size gives a time-varying modification. Due to the separation of the stochastic and deterministic sound components, this representation is more successful in time-scale modifications than traditional additive synthesis techniques. The noise part of the sound remains noise no matter how much the sound is stretched (which is not the case in "sinusoids-only" representations).

In the deterministic representation, each function pair, amplitude and frequency, accounts for a partial of the original sound. The manipulation of these functions is easy and musically intuitive. All kinds of frequency and magnitude transformations are possible. For example, the partials can be transposed in frequency, with different values for every partial and varying during the sound. It is also possible to decouple the sinusoidal frequencies from their amplitude, obtaining effects such as changing pitch while maintaining formant structure.

The stochastic representation is modified by changing the shape of each of the envelopes. Changing the envelope shape corresponds to further filtering of the stochastic signal. Their manipulation is much simpler and more intuitive than the manipulation of a set of all-pole filters, such as those resulting from an LPC analysis.

Interesting effects can also be accomplished by changing the relative amplitude of the two components, thus emphasizing one or the other at different moments in time.

The characterization of a single sound by two different representations may cause problems. When different transformations are applied to each representation, it is easy to create a sound in which the two components, deterministic and stochastic, do not fuse into a single entity. This may be desirable for some musical applications, but in general it is avoided and requires some practical experimentation with the actual representations.

The best synthesis is generally considered the one that results in the best perceptual identity with respect to the original sound. Then, transformations are performed on the corresponding representation. For musical applications, however, this may not always be desirable. Very interesting effects result from purposely setting the analysis parameters "wrong." We may, for example, set the parameters such that the deterministic analysis only captures partials in a specific frequency range, leaving the rest to be considered stochastic. The result is a sound with a much stronger noise component.

Conclusion

Spectral modeling synthesis is an analysis-based technique capable of capturing the perceptual characteristics of a wide variety of sounds. The representation that results from the analysis is intuitive and is easily mapped to useful musical parameters.

The analysis part is central to the system. It is a complex algorithm that requires the manual setting of a few control parameters. Further work may automate the analysis process, particularly if there is a specialization for a group of sounds. Some aspects of the analysis are also open to further research, in particular the peak-continuation algorithm.

The synthesis from the deterministic plus stochastic representation is simple and can be per-

formed in real-time with current technology. A real-time implementation of this system would allow the use of this technique in performance. The representation would be computed ahead of time and stored, and the sound transformations would be done interactively.

References

Allen, J. B. 1977. "Short-term Spectral Analysis, Synthesis, and Modification by the Discrete Fourier Transform." *IEEE Transactions on Acoustics, Speech, and Signal Processing* 25(3): 235–238.

Allen, J. B., and L. R. Rabiner. 1977. "A Unified Approach to Short-time Fourier Analysis and Synthesis." *Proceedings of the IEEE* 65(11): 1558–1564.

General Electric Co. 1977. "ADEC Subroutine Description." Heavy Military Electronics Dept., Syracuse, New York, Document number 13201, June 1977.

Grey, J. M. 1975. "An Exploration of Musical Timbre." Ph.D. diss., Stanford University.

Harris, F. J. 1978. "On the Use of Windows for Harmonic Analysis with the Discrete Fourier Transform." *Proceedings of the IEEE* 66(1): 51–83.

Hedelin, P. 1982. "A Representation of Speech with Partials." In D. Carlson and B. Granstrom, eds. *The Representation of Speech in the Peripheral Auditory System.* Elsevier Biomedical Press, Amsterdam, Netherlands.

McAulay, R. J., and T. F. Quatieri. 1984. "Magnitude-only Reconstruction using a Sinusoidal Speech Model." *Proceedings of the 1984 IEEE International Conference on Acoustics, Speech, and Signal Processing.*

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

Markel, J. D., and A. H. Gray. 1976. *Linear Prediction of Speech.* New York: Springer-Verlag.

Moorer, J. A. 1973. "The Heterodyne Filter as a Tool for Analysis of Transient Waveforms." Memo AIM-208. Stanford Artificial Intelligence Laboratory, Computer Science Dept., Stanford University.

Moorer, J. A. 1977. "Signal Processing Aspects of Computer Music—A Survey." *Computer Music Journal* 1(1): 4–37.

Moorer, J. A. 1978. "The Use of the Phase Vocoder in Computer Music Applications." *Journal of the Audio Engineering Society* 26(1/2): 42–45.

Portnoff, M. R. 1976. "Implementation of the Digital Phase Vocoder Using the Fast Fourier Transform." *IEEE Transactions on Acoustics, Speech, and Signal Processing* 24(3): 243–248.

Rife, D. C., and R. R. Boorstyn. 1974. "Single-Tone Parameter Estimation from Discrete-Time Observations." *IEEE Trans. Info. Theory* 20:591–598.

Rife, D. C., and R. R. Boorstyn. 1976. "Multiple Tone Parameter Estimation from Discrete-Time Observations." *Bell Systems Tech. Journal* 155:1389–1410.

Serra, X. 1989. "A System for Sound Analysis/Transformation/Synthesis Based on a Deterministic plus Stochastic Decomposition. Ph.D. diss., Stanford University.

Smith, J. O., and B. Friedlander. 1984. "High Resolution Spectrum Analysis Programs." Memo 5466-05. Systems Control Technology, Palo Alto, California.

Smith, J. O., and X. Serra. 1987. "PARSHL: An Analysis/Synthesis Program for Nonharmonic Sounds based on a Sinusoidal Representation." *Proceedings of the 1987 International Computer Music Conference.* San Francisco: Computer Music Association.

Strawn, J. 1980. "Approximation and Syntactic Analysis of Amplitude and Frequency Functions for Digital Sound Synthesis." *Computer Music Journal* 4(3): 3–24.

Wolcin, J. J. 1980a. "*Maximum A Posteriori Line Extraction: A Computer Program.*" USC Technical Memorandum 801042, March 20.

Wolcin, J. J. 1980b. "Maximum A Posteriori Estimation of Narrowband Signal Parameters." USC Technical Memorandom 791115, June 21, 1979, *Journal of the Acoustical Society of America* 68(1): 174–178.