Analysis/Synthesis Comparison

Matthew Wright (CNMAT), James Beauchamp (UIUC), Kelly Fitz (CERL Sound Group), Xavier Rodet (IRCAM), Axel Röbel (CCRMA, now IRCAM), Xavier Serra (IUA/UPF), Gregory Wakefield (UM)

matt@cnmat.berkeley.edu, j-beauch@uiuc.edu, kfitz@cerlsoundgroup.org, rod@ircam.fr, Axel.Roebel@ircam.fr, xserra@iua.upf.es, ghw@eecs.umich.edu

CNMAT: University of California at Berkeley, 1750 Arch St., Berkeley, CA 94709, USA
UIUC: School of Music and Electrical and Computer Engineering, University of Illinois at Urbana-Champaign, Urbana, IL 61801 USA
CERL Sound Group: c/o Prof. Lippold Haken, UIUC
IRCAM: 1 Place Igor Stravinsky, 75004 Paris, France
CCRMA: Dept. of Music, Stanford University, Stanford, CA 94305-8180, USA
IUA/UPF: Audiovisual Inst., Pompeu Fabra Univ., Rambla 31, 08002 Barcelona, Spain
UM: University of Michigan EECS Dept.,1101 Beal Avenue, Ann Arbor, MI 48109, USA

Abstract

We compare six sound analysis/synthesis systems used for computer music. Each system analyzed the same collection of 27 varied input sounds, and output the results in Sound Description Interchange Format (SDIF). We describe each system individually then compare the systems in terms of availability, the sound model(s) they use, interpolation models, noise modeling, the mutability of various sound models, the parameters that must be set to perform analysis, and characteristic artifacts. Although we have not directly compared the analysis results among the different systems, our work has made such a comparison possible.

Introduction: Analysis/Synthesis

Sound analysis/synthesis has a rich history in both music and speech (Risset and Mathews, 1969, Schafer and Rabiner, 1973, Crochiere, 1980, McAulay and Quatieri, 1985), as Risset and Wessel describe in their review (Risset and Wessel, 1999). For our purposes, "analysis" means the selection of a sound model and the estimation of its parameters for a given input sound[1], and "synthesis" is the process of turning that model (or a modified model) back into sound.

There are three goals of analysis/synthesis. The first is to reproduce the perceptual features of the input sound; if the resynthesis is indistinguishable from the original input sound then the sound model clearly captures the sound (Wakefield, 2000). We might distinguish two ways to fall short of this goal: leaving out perceptual features of the input in the resynthesis, and adding new perceptual features (often called "artifacts") to the resynthesis that were not in the input.

The second goal of analysis/synthesis is low dimensionality of the model. Sometimes an analysis procedure performs data reduction, where the result of analysis is represented with fewer bits than the original sound; this has obvious practical applications. Such data reduction

[1] In the computer music field we represent the "input sound" as PCM time-domain digital samples, which is itself a model. In this paper we will ignore this point, treating PCM samples as the beginning and end of the analysis/synthesis story.

can be obtained through standard statistical methodologies; however, a more interesting case is when the low dimensionality of the model is based on perceptual or production criteria. In such cases, not only is there a reduction in data, but greater insight is provided with respect to the perceptual features of the sound or to how the sound was generated by the performer.

The third goal is related to the second: analysis/synthesis should afford musically interesting transformations and control. Musicians should be able to alter the parameters of the analysis result to produce new sounds related to the original input sound. Ideally, models based on the results of analysis should provide for control, and even real-time control. The common transformations that have been performed historically with analysis/synthesis of musical sounds are time-scale modification, pitch modification, and cross-synthesis.

Motivation for this Comparison

The typical technical paper describing an analysis/synthesis technique, if it includes sound examples, generally presents sounds that show the technique in its best light, both in terms of choosing input sounds and in terms of choosing transformations to produce new sounds. Musicians and composers inspired to use a given analysis/synthesis package often discover that it does not work as well as the published examples. This makes it difficult to compare analysis/synthesis techniques by reading papers: a paper will show a technique's strengths but not its weaknesses, and often does not give guidance in choosing which technique to use in which situation.

Another difficulty in comparing analysis/synthesis techniques is the lack of a standard database of input sounds. The speech community has many such databases for evaluating speech compression algorithms (LDC, 2000). Adrian Freed proposed such a database for evaluation of fundamental frequency estimators (Freed and Jehan, 1997).

These difficulties motivated the *Analysis/ Synthesis Comparison* panel session at the 2000 International Computer Music Conference in Berlin. The goal of the panel session was to make a meaningful comparison among the major analysis/synthesis techniques and software packages used for computer music. This comparison was not a competition, but an attempt to learn more about currently available analysis/ synthesis systems and how best to use them.

Two factors allowed the comparison to be meaningful: the common input sounds and the common output format. Each participant analyzed the same set of 27 input sounds. The chosen output format for this comparison was the Sound Description Interchange Format ("SDIF") (Wright, et al., 1998, Wright, et al., 1999, Schwarz and Wright, 2000). SDIF has now become a standard for the representation of analysis results in this field. Before SDIF, analysis/synthesis systems could be compared based on resynthesized sound output only; with SDIF, it is now possible to compare the analysis results themselves from different systems.

What We Did

Matthew Wright conceived this panel session and invited potential panelists. Although most computer music analysis/synthesis tools were represented at this session, we recognize that several important models were underrepresented, for example, wavelets (Kronland-Martinet, 1988). We hope that our work has made it possible to add other analysis/synthesis systems to the comparison in the future.

Each panelist downloaded the collection of input sound files, ran them through their analysis software package, saved the results as SDIF

files, and made the SDIF files publicly available. Panelists also shared sound files containing their resynthesized input sounds as well as resynthesized transformations of the input sounds. All these results, as well as the original input sounds, this paper, links to all of the URLs mentioned in this paper, and other information about the comparison are available at http://www.cnmat.berkeley.edu/SDIF/ICMC2000

The structure of this paper mirrors that of the two-hour panel session in Berlin: an introduction to analysis/synthesis and to each individual software package, followed by a guided group discussion focussing on specific points of comparison and on questions raised by the comparison.

Most panelists added SDIF support to their software in the course of participating in the panel. We also extended SDIF to support some panelists' software, or, more specifically, to support the sound models used by some panelists' software.

The Participants

This comparison includes six analysis/synthesis systems:

- SNDAN (Beauchamp, 1993), represented by James Beauchamp (University of Illinois at Urbana-Champaign)
- SMS (Serra and Smith, 1990), represented by Xavier Serra and Maarten de Boer (Music Technology Group, Audiovisual Institute, Pompeu Fabra University)
- PartialBench (Röbel, 1999), represented by Axel Röbel (CCRMA)
- Loris (Fitz, et al., 2000b, Fitz, et al., 2000a), represented by Kelly Fitz and Lippold Haken (CERL Sound Group)
- IRCAM's analysis/synthesis suite (Rodet, 1997a, Rodet, 2000), represented by Xavier Rodet and Diemo Schwarz (IRCAM)
- MDRx (Pielemeier and Wakefield, 1996, Mellody and Wakefield, 2000b), represented by Gregory Wakefield (University of Michigan)

The Input Sounds

The participants in the panel contributed the input sounds for this comparison. All of these sounds appear as sound examples on the CD accompanying this journal. Twenty-seven sounds were selected for variety and brevity, and consisted of the following:

- Two single-note examples from trumpet and piano
- Eight instrumental monophonic musical phrases, including saxophone, noisy shakuhachi and suling flutes, a highly reverberant clarinet, flutter-tongued trombone, and single-note lines played on piano and steel-string guitar.
- Five percussive sounds varying in noisiness and inharmonicity: single notes from xylophone, Gamelan gong and bass drum, and longer rhythms from berimbao and bongo.
- Two polyphonic instrumental examples: tamboura (with multiple unison strings and a low octave) and a harp glissando.
- Five sung examples: an isolated dry single voice, yodeling, multiple voices in unison, a polyphonic chorus, and singing into a flute while playing it.
- Two speech examples, including a giggle
- The noisy sound of biting into an apple
- The scream of an angry cat
- One purely synthetic sound: 10 simultaneous chirps crossing in frequency

Sound Models

The key to understanding analysis/synthesis lies in understanding the "sound models" whose parameters are fit to the input sound in the analysis process. Finding the differences in the models used by each participant's software was one of the main goals of the panel. The potential interoperability of the participants' different analysis/synthesis systems is limited only by the differences in the underlying sound models.

This section briefly introduces the various sound models used by the participants' analysis/synthesis systems; the individual sections that follow give more detail on each system.

Sinusoidal models (Helmholtz, 1875) were heavily represented in this panel, but even within this category, each participant had different sound models. Some sinusoidal analysis/synthesis programs are specifically geared towards harmonic series, (SNDAN's *pvan* and IRCAM' s *additive*), while others use a more general sum-of-sinusoids model with no restriction on frequency (PartialBench, SNDAN's *mqan*, IRCAM's *hmm*, Loris, MDRx, and SMS). Sinusoidal models also differ in the temporal sampling of frequency, amplitude, and phase envelopes and in the assumptions about how to use interpolation to produce these envelopes from the discretely sampled values. (See the section "Interpolation of Sinusoidal Parameters" below.)

Loris uses "reassignment" (Auger and Flandrin, 1995) to place each individual frequency/amplitude sample pair at a unique place in time, unlike all the other additive models, which group parameter estimates together (into a "frame") at a single time for all of the partials. Another difference is that Loris' additive partials are not pure sinusoids, but mixtures of sinusoidal and noise energy, using the "Bandwidth-Enhanced Additive" model (Fitz and Haken, 1995).

SMS and IRCAM's sinusoidal modeling programs represent noise with a sinusoidal plus residual model. In this model, part of each sound is represented with a sinusoidal model, and then the rest of the sound (the "residual," the remainder of subtracting the resynthesized sinusoids from the original signal) is represented separately, for example, with spectrally shaped white noise. (See the section "Modeling of Noise" below.) SMS also allows the entire sound to be treated as a residual, skipping the sinusoidal analysis and representing everything with the noise model.

Rational models provide an alternative to additive synthesis, and are known by a variety of different names including all-pole, pole-zero, auto-regressive moving average (ARMA), LPC, or formant synthesis. In general, these models assume the signal $y(n)$ obeys the following equation:

$$y(n) = \sum_{k=1}^{p} a_k y(n-k) + \sum_{k=0}^{z} b_k x(n-k)$$

where $x(n)$ is an input signal and the coefficients $\{a_k\}$ and $\{b_k\}$ characterize a linear time-invariant system. This type of signal model has a long and rich history beginning with the statistical work of time-series modelers in the 1920's who coined the terms *autoregressive* (AR) and *moving average* (MA) to refer to the summations involving $\{a_k\}$ and $\{b_k\}$, respectively. When viewed as the input-output relationship of a linear system, the transfer function is ratio of two polynomials (a rational model) whose coefficients in the denominator are the $\{a_k\}$ and those in the numerator are the $\{b_k\}$. Alternatively, the poles and zeros of the transfer function are determined by the $\{a_k\}$ and $\{b_k\}$, respectively.

The systems in this equation are static over time so that any type of temporal variation must be modeled by the input signal. Interest in pole-zero models for dynamically varying systems increased in the late 1960's and early 1970's in part because of the research of Fant concerning the acoustics of speech over short time periods (e.g., 5-10 milliseconds). He argued that poles could model the vocal tract, for all but nasal speech sounds, in which case the zeros were also required. The speech synthesis community adopted the term *Linear Predictive Coding* (LPC) to refer to synthesis algorithms based on all-pole models of the speech signal, in which case the $\{a_k\}$ become LPC coefficients. In the music-synthesis community, formant synthesis is a method for efficiently synthesizing signals that obey the above equation when p and z are relatively small and the input signal x can be constructed as a periodically repeated waveform.

MDRx uses a non-Fourier-based time-frequency representation (Cohen, 1995), the Modal Distribution. This representation is further analyzed to produce the parameters of sinusoidal, resonance, and pole/zero models.

IRCAM's suite of analysis/synthesis tools include sinusoidal models, linear prediction analysis, autoregressive lattice filter, fundamental frequency estimates, voiced/unvoiced detection (Peeters and Rodet, 1998), pitch-synchronous marker finding (Peeters and Rodet, 1999), resonance modeling (Potard, et al., 1986), FOF synthesis (Rodet, et al., 1984, Rodet, et al., 1987), and transient detection.

SDIF and Sound Models

SDIF is a general-purpose sound description format framework. Each data element (a frame or a matrix) has a type identifier, and each kind of sound description, i.e., each different kind of sound model, is represented in SDIF by frames and matrices of appropriate types. There is a library of standard SDIF frame and matrix types, and the ability to develop and eventually standardize new types.

SDIF's collections of standard frame and matrix types were expanded to accommodate the sound models used by participants in this panel. We added a representation for the Modal Distribution, and backwards-compatibly upgraded the sinusoidal track models to support both reassignment and bandwidth enhancement. IRCAM has proposed new SDIF types (http://www.ircam.fr/anasyn/sdif/standard/types-main.html) for LPC, autoregressive lattice filter, FOFs, and time markers.

The subtle aspect of extending SDIF to incorporate each participant's sound models was to distinguish the sound model from the analysis method used by each program. One lazy extreme would be to define new SDIF frame and matrix types for each program; this would cause needless incompatibility. One way we avoided this was by using the standard "1TRC" ("sinusoidal track") type to represent the sinusoidal models output by different software.

The opposite extreme would be to disregard the differences in the sound models, forcing semantically different sound models into a single representation; this would cause a loss of information and a weakening of the meaning of SDIF's model types. One way we avoided this was by defining a new type for the Modal Distribution model, rather than attempting to use the existing "1STF" (Short-Term Fourier Transform) types.

SNDAN

SNDAN (Beauchamp, 1993) is a software package for analysis, graphics, modification, and resynthesis for musical applications. SNDAN consists of a suite of command-line driven programs for any Unix workstation; there is also a working version for Windows/DOS available

from the United Kingdom Composers Desktop Project. SNDAN was developed by James Beauchamp, Robert Maher, George Chaltas, Timothy Madden, Jonathan Mohr, and others at the University of Illinois at Urbana-Champaign.

SNDAN is freeware and available for download as ANSI C source code (see http://ems.music.uiuc.edu/~beaucham/software/sndan). Users who can program in C are encouraged to make their own modifications and install their own special commands if they so desire.

SNDAN provides two different methods of analysis based on the Fast Fourier Transform ("FFT"). The first, called *pvan*,is a fixed filterbank tuned phase-vocoder (Portnoff, 1976, Moorer, 1978, Dolson, 1986) approach, where frequency bins are harmonically related. The other is a frequency tracking method based on a technique developed by McAulay and Quatieri in the 1980's (McAulay and Quatieri, 1986), called *mqan.*

With *pvan*, the user specifies the fundamental frequency of analysis. Using sample-rate conversion, a Hamming-windowed FFT over twice the inverse of this frequency yields a series of filters with center frequencies at harmonics of the fundamental frequency, and zero responses at the other harmonics. Thus, at least for steady-state tones, with a properly tuned analysis, the output of each filter corresponds to the amplitude of the harmonic sine wave that falls with the range of that filter. Moreover, the frequency deviation from the band center of any mistuned harmonic is computed using the phase output of the corresponding filter.

The result of a SNDAN *pvan* analysis is the spectral magnitude and frequency of each harmonic bin output. These values are sampled in frames; the interframe time is one-fourth of the FFT window size, which is one-half of the fundamental period. Since the input sound is assumed to be harmonic, each partial's inter-frame identity comes from its harmonic partial number. Only the initial phase is given for each partial. Since the fundamental frequency is assumed to be constant, the number of partials is the same in each frame. More details are given in (Beauchamp, 1993).

With *mqan*, the user specifies a minimum analysis frequency and a threshold value. The FFT window size is chosen such that the minimum frequency is well resolved. A Kaiser window with alpha = 2 and a 100% zero-fill factor is used. In each frame, *mqan* identifies all spectral magnitude peaks above the given threshold; these are assumed to correspond to sinusoids in the signal. The amplitude and frequency of each peak are refined by parabolic interpolation.

The result of a SNDAN *mqan* analysis is the frequency, amplitude, and phase of each partial. Each frame may have a variable number of partials. The partials of each frame are joined to partials of the following frame to form "tracks"; if no match is found for a partial, then that partial "expires." The interframe time for *mqan* analysis files is currently set to 128 samples, although this is easy to change. Papers by McAulay and Quatieri (McAulay and Quatieri, 1986), Smith and Serra (Smith and Serra, 1987), and Maher (Maher, 1990) give more details.

SNDAN includes a method for converting from *mqan* sound models to *pvan* sound models. A pitch detector optimized for multi-pitched unaccompanied solo recordings determines the fundamental frequency versus time function for an *mqan* sound model (Maher and Beauchamp, 1994). A second program produces a *pvan* sound model from the *mqan* sound model and its fundamental frequency envelope.

SNDAN includes two programs for displaying analysis data. Visual representations for *mqan* sound models are restricted to 2D (frequency vs. time tracks) and 3D (amplitude vs. time vs. fre-

quency) graphs. Visual representations for *pvan* sound models include a large choice -- currently 15 -- of graph types, which afford different ways of looking at harmonic amplitude and frequency data. For example, “brightness” (spectral centroid) and “spectral irregularity” can be plotted as functions of time.

While SNDAN is strictly command-line driven and non-real-time, the SNDAN team has developed Armadillo (http://ems.music.uiuc.edu/~beaucham/software/armadillo) for the Power Macintosh which performs real-time or non-real-time analysis on microphone, CD-ROM, or sound file input (Madden and Beauchamp, 1999). Armadillo has a graphical user interface and a variety of real-time display features. Armadillo is an outgrowth of SNDAN, and is available separately.

SMS

SMS (“Spectral Modeling Synthesis” http://www.iua.upf.es/~sms) is a set of techniques and software implementations for the analysis, transformation and synthesis of musical sounds. SMS started as Xavier Serra’s doctoral dissertation at Stanford University (Serra, 1989), and is now developed by the members of the Music Technology Group of the Audiovisual Institute at Pompeu Fabra University (UPF): Jordi Bonada, Maarten de Boer, Eduard Resina, Xavier Serra, and others. SMS runs under Windows, Linux, and MacOS and is available only in binary form.

SMS is based on the idea that each sound and application may require a different model, thus the user should be able to specify the model to use and to control all the relevant parameters of the model. The model for which SMS was originally developed assumes that a sound can be decomposed into a deterministic plus a stochastic part (Serra and Smith, 1990). In this model the deterministic part is represented with sinusoids that have time-varying frequencies and amplitudes, and the stochastic part is represented either by filter coefficients with a gain value or by spectral magnitudes and phases. SMS now also provides two other models that are building blocks of that one: the short-time Fourier transform and the sinusoidal model. From the deterministic plus stochastic representation the analysis also extracts and stores other higher level attributes of the sound, such as fundamental frequency, spectral shape, spectral tilt, or note boundaries.

SMS includes several programs, with command line and graphical interfaces, for analyzing and synthesizing sounds for both real-time and non-real-timework.

PartialBench

PartialBench is a Matlab program written by Axel Röbel while at the Electronic Studio of the Technical University of Berlin and later as a visiting scholar at CCRMA. It extends an earlier adaptive approach (Röbel, 1999) and provides improved spline based models for amplitude and phase trajectories with freely selectable polynomial order and an improved representation of resonators that are to close in frequency to be resolved into single partials, as for example in percussive sounds. As a result the representation is much more robust with respect to signal modifications. The current version is still under development and not yet publicly available.

The main idea of PartialBench is to find sinusoidal partials that are related to the physical resonances that constitute the sound. The partial parameters are described by means of piecewise polynomial functions expressed by means of B-Splines. By using splines it is possible to express each piecewise parameter trajectory as a superposition of basic functions and by means of changing the basis functions the degree of the polynomial and the smoothness of the trajectory

can easily be changed. The model is initialized using an FFT-based analysis. The weighting parameters of the spline basis functions are adaptively adjusted such that the error between the original signal and partial model is minimized. The adaptive procedure is computationally much more expensive than the standard FFT based approaches. However, it comes with the advantage, that a complete and continuous model of the amplitude and phase trajectories is provided and no heuristics are needed to resolve for example any contradictions between the sampled frequency and phase estimates.

For modeling real-world sound signals, two cases have to be distinguished: either single resonances can be resolved into single partials, or the resonances are too dense to be resolved and a single partial has to be used to model a group of resonances. Both cases can be handled and result in partial models that can be used for high quality signal modifications such as time stretching and transposition.

Using splines, a parameter trajectory model is defined by means of the position of the basis functions and the order of the polynomial. It can be shown that the positions of the basis functions, which are the break point positions of the spline, are mainly responsible for defining the frequency resolution of the analysis (the further apart the basis functions the higher the resolution and the smaller the maximum band width of the partial). The order of the polynomial affects the side lobe suppression of the analysis (the higher the order the less the analysis results are affected by out of band energy).

The current version of PartialBench is still mainly devoted to research, and is intended to investigate whether additive synthesis can be used to establish models with physical interpretation for different classes of non-stationary sounds (for example all types of percussive sounds).

Loris

Loris (http://www.cerlsoundgroup.org/Loris) is a C++ class library implementing analysis, manipulation, and synthesis of digitized sounds using the Reassigned Bandwidth-Enhanced Additive Sound Model (Fitz, et al., 2000a, Fitz, et al., 2000b). Loris itself is not an application program but a suite of procedures that can be linked into an application or called from a scripting language. Currently, Loris has been wrapped in extension modules for Python, Tcl, and Perl, and has been tested under Linux, IRIX 6.5, and MacOS 9.

Loris was developed by Kelly Fitz and Lippold Haken at the CERL sound group. Loris is covered by the Gnu Public License; see http://www.sourceforge.net for source code.

The Reassigned Bandwidth-Enhanced Additive Model shares with traditional sinusoidal models the notion of temporally connected partial parameter estimates. By contrast, reassigned estimates are non-uniformly distributed in both time and frequency, yielding greater resolution in time and frequency than is possible using conventional additive techniques. Partial parameter envelopes are obtained by following ridges on a time-frequency surface. Loris uses the method of reassignment (Auger and Flandrin, 1995) to improve the time and frequency estimates for the envelope breakpoints.

Bandwidth enhancement expands the notion of a partial to include the representation of both sinusoidal and noise energy by a single component type (Fitz and Haken, 1995). Loris represents each reassigned bandwidth-enhanced partial with a trio of synchronized breakpoint envelopes specifying the time-varying amplitude, center frequency, and noise content (or bandwidth) for each component. The bandwidth envelope allows each component to represent a mixture of sinusoidal and noise energy. Bandwidth asso-

ciation is the process of constructing these bandwidth envelopes, or in other words, determining how much noise energy should be represented by each bandwidth-enhanced partial. Loris synthesizes bandwidth-enhanced partials using a stochastic amplitude-modulation technique for spectral line widening.

The time-variant parameters of bandwidth-enhanced partials can be used to manipulate both sinusoidal and noisy components of sound in an intuitive way, using a familiar set of controls. The encoding of noise associated with a bandwidth-enhanced partial is robust under partial parameter transformations, and is independent of other partials in the representation. Bandwidth-enhanced partials can be modified without destroying the character of noisy sounds or introducing audible artifacts related to the representation of noise.

IRCAM's Analysis/Synthesis Software

IRCAM's Analysis/Synthesis team has produced a large suite of analysis/synthesis tools (http://www.ircam.fr/anasyn), including sinusoidal models, linear prediction analysis, autoregressive lattice filter, fundamental frequency estimates, voiced/unvoiced detection, pitch-synchronous marker finding for pitch-synchronous overlap/add (PSOLA) synthesis (Peeters and Rodet, 1999), resonance modelling (Potard, et al., 1986), FOF synthesis (Rodet, et al., 1984, Rodet, et al., 1987), spectral envelopes (Schwarz and Rodet, 1999), and transient detection. Authors include P.F. Baisnée, P. Chose, P. Depalle, B. Doval, G. Garcia, F. Iovino, F. Jaillet, F. Marti, G. Peeters, G. Poirot, Y. Potard, S. Roux, D. Schwarz, X. Rodet, and D. Virolle. Supported platforms include IRIX, DEC-OSF, Linux, and MacOS; not every tool is implemented for each platform. Most of this software is available for an annual fee from the IRCAM Forum (http://www.ircam.fr/forum).

The *additive* program (Doval and Rodet, 1993, Rodet, 1997b) uses a model of sinusoidal partials plus a noisy residual, modeling the residual with spectral envelopes, with the assumption that the input sound is harmonic and monophonic. A "harmonic sieve" looks for the *n*th harmonic partial in each frame by finding the loudest spectral peak within a given frequency range around *n* times the fundamental frequency.

IRCAM promotes the technique of running *additive* twice, first on the input sound, then on the residual created by subtracting the first resynthesis from the original. This is useful if the file is complex and contains two different harmonic sounds of different fundamental frequency.

The *hmm* program (Depalle, et al., 1993) also uses a sinusoidal model, but makes no assumptions about harmonicity, instead using Hidden Markov Models to link spectral peaks at different times into continuous tracks. Both *additive* and *hmm* run on Unix and Macintosh and have interfaces on the Macintosh through *Diphone*.

The *Chant* program uses a FOF plus filter model, and the modRes program uses a resonance model (i.e., resonant filters or sinusoids with exponentially decaying amplitude implemented as FOFs). Again, these are Unix and Macintosh programs with a *Diphone* interface for Macintosh.

MDRx

MDRx is a software package that implements the Modal Distribution. MDRx was developed by Gregory H. Wakefield, Maureen Mellody, Andrew Sterian, Rowena Guevara, William Pielemeier, and Anastasia Yendiki at the MusEn Project at the University of Michigan. MDRx is implemented as a Matlab script and as a Win-

dows 98/NT application. See http://www.eecs.umich.edu/~ghw for availability.

The Modal Distribution ("MD") is a bilinear time-frequency distribution designed for sounds dominated by partials. The mathematical consequence of partials is that the theoretical time-frequency surface of such signals is relatively sparse, with energy concentrated primarily within the neighborhoods of each partial. In a variety of test cases, the instantaneous frequencies and amplitudes of time-varying partials can be determined with high degrees of accuracy, typically exceeding those achieved by other analysis methods.

MD analysis requires the user to set only a single parameter, the minimum frequency separation between partials. For a single-voice harmonic sound, this is the lowest fundamental frequency over the musical passage of interest. The Modal Distribution takes advantage of the sparse time-frequency surface in a way that permits the nominal minimum frequency separation to differ from the true minimum separation by as much as a half octave without substantial error. This is in contrast to techniques based on Fourier analysis, especially the phase vocoder, which are often very sensitive to errors in the nominal fundamental frequency.

MD analysis is substantially more expensive computationally than many other methods, although with current microprocessors real-time ratios between 10:1 and 20:1 are typical for frame rates of 4 milliseconds, analysis windows of 2-4k samples, and fundamentals as low as 80 Hz. The computation time is inversely proportional to the fundamental frequency.

Several forms of synthesis are consistent with the sparse structures MD analysis assumes. Additive synthesis is the most common. In this case, ridges along the time-frequency surface are interpreted as partials and the local moments of these ridges are used to determine the instantaneous frequency and amplitude of each partial. This surface information can also be used to fit the parameters required in pole/zero-source modeling, such as formant synthesis or LPC.

Successful applications have included flutter-tongued flute (Pielemeier, 1993), piano (Guevara and Wakefield, 1996, Guevara, 1997), violin (Mellody and Wakefield, 2000b), trumpet (Mrozek, 1999), the singing voice (Mellody, 2000, Mellody, et al., 2000), and spasmodic dysphonia (Wakefield, et al., 2000). In each of these cases the MD model was then fit to an additive and/or pole/zero model.

In its present form, the linkage between MD analysis and MD synthesis remains weak. Details in the surface, which would normally be smoothed over by alternative analysis methods, must be extracted algorithmically. In addition, we take no explicit advantage of prior knowledge concerning the pitch. Such tracking problems are endemic to most analysis/synthesis methods, and algorithms are being developed to extract the relevant features automatically from the time-frequency surfaces. The work discussed above reflects the outcome of longer-term research projects, rather than the results of a single-pass algorithm. We have considered simpler versions of the automatic synthesis question. Multi-voice transcription, a weaker type of tracking problem, has been studied using MD analysis and multiple-target tracking. Transcription results for samples from a brass quartet support the utility of this approach (Sterian and Wakefield, 1998, Sterian, 1999, Sterian, et al., 1999).

MDRx's best results in this comparison drew from samples that were dominated by partials in relatively non-reverberant environments. Automatic synthesis of the voice and horn samples, in particular, was successful, with the exception

that non-voiced materials led to a buzz-artifact during synthesis. Regressing to models of decaying exponentials led to improved quality in the synthesis of several of the percussive tracks. MDRx had a more difficult time handling multi-timbral materials with relatively long decay times, such as the harp. Highly noisy sources were also harder to reproduce, although the flute and voice/flute materials were synthesized reasonably well.

The MDRx team has spent far less time on issues of mutability and transformations of synthesis parameters, in part, because our focus has been primarily on highly detailed analysis and modeling of acoustic instruments. A different study reports on morphing the identities of female singers based upon MD (Mellody and Wakefield, 2000a).

Summary Comparison of Each System

This table lists the availability of each analysis/synthesis system:

	SNDAN	SMS	PartialBench	Loris	IRCAM's	MDRx
Open-Source?	√			√		
Freely Available?	√	√		√		√
Costs money?					√	

This table lists the form in which each analysis/synthesis system is embodied. No distinction is made between a single application program and a suite of application programs:

	SNDAN	SMS	PartialBench	Loris	IRCAM's	MDRx
Application	√	√			√	√
C++ Library				√		
Python, Tcl, Perl modules				√		
Matlab script			√			√

This table lists the platforms supported by each analysis/synthesis system. We consider Matlab to be a platform, since a Matlab program can run on any hardware/OS combination for which Matlab is implemented.

	SNDAN	SMS	PartialBench	Loris	IRCAM's	MDRx
Linux	√	√		√	√	
Other Unix	√			√	√	
Macintosh		√		√	√	
Windows		√		√		√
Matlab			√			√

This table lists the kinds of sound models that can be output by each analysis/synthesis system:

	SNDAN	SMS	PartialBench	Loris	IRCAM's	MDRx
Fundamental frequency envelope	√	√			√	√
Harmonic sinusoidal model	√	√			√	√
Sinusoidal model	√	√	√	√	√	√
Bandwidth-enhanced sinusoidal model				√		
Separate residual		√			√	

How Each System Produced SDIF

IRCAM and CNMAT each have a publicly available SDIF library that handles the details of reading and writing SDIF data (http://www.ircam.fr/sdif and http://www.cnmat.berkeley.edu/SDIF). IRCAM's library is released under the Gnu GPL and soon under the Gnu LGPL.

IRCAM has integrated SDIF into its entire suite of analysis/synthesis software, and now all programs use SDIF by default for input and output. IRCAM has provided a Matlab extension to read and write SDIF files, which has been used by PartialBench and MDRx as well as within IRCAM.

PartialBench's spline-based trajectory model is currently not well supported by SDIF, because SDIF lacks a representation for interpolation models. (See "future work.") To support interchange with other synthesizers, PartialBench recalculates its spline-based third order parameter trajectory models in terms of a densely sampled (around 10 milliseconds) first order model for SDIF output. Because the number of basis functions of the linear model is much larger than the original number of basis functions, this re-

sampling increases the model size by about one order of magnitude.

Loris has been modified to incorporate CNMAT's SDIF library and now uses SDIF as its only file format for analysis results. SDIF's data types for sinusoidal track models (the "1TRC" frame and matrix types) were extended to support Loris' bandwidth-enhancement and reassignment. We wanted to facilitate interchange of sound models output from Loris with other programs that do not implement bandwidth-enhancement and/or reassignment. In particular, we wanted it to be easy for a program to ignore the Loris-specific part of a model and still use the standard frequency, amplitude, and phase values for each partial. This was accomplished by defining optional matrix columns. An SDIF file from Loris contains the standard parameters for each partial in the first columns, followed by bandwidth (ranging from 0, a pure sinusoid, to 1, pure band-limited noise) and a time offset (a delta time relative to the frame's time) in extra columns.

The SMS software can import and export a large part of the SMS analysis data as SDIF files, using the SDIF standard frame types for Fundamental Frequency Estimates, Short-Term Fourier Transform and Sinusoidal Tracks. As the native SMS file format already contains this data, the translation is very straightforward, despite of the fact that the original SMS data is stored in frames that contain all data, rather than separate frames per data type. The *SDIFDisplay* program, developed at the UPF, can be used to display SDIF files output from SMS. Both SMS and *SDIFDisplay* make use of C++ classes written on top of the CNMAT SDIF library. Future extensions of using SDIF within SMS could be a support for the various SMS high level attributes, regions, and indicating the relationship between the frame types.

SNDAN uses custom file formats for the results of *pvan* and *mqan* analyses, which can be converted to SDIF without loss of information. Matthew Wright wrote SNDAN to SDIF conversion programs that are available online with the other results of the comparison. SDIF's sinusoidal track model is more general than SNDAN's, because SNDAN assumes regular time-sampling of partial frames and SDIF allows the interframe time to vary arbitrarily. Converting SDIF data to SNDAN in the general case therefore requires interpolated resampling of partial parameters.

SNDAN's custom file formats include the parameters of the analysis: sample rate, duration, maximum amplitude, analysis fundamental frequency, frame duration, FFT length, number of harmonic bins, number of channels (for the future), number of frames, format type, and date of analysis. SNDAN files also include information about the original input sound: performer, instrument, date recorded, musical pitch, dynamic, vibrato (yes/no), part of sound, comments. This information can be represented with SDIF's name/value tables.

Choosing the Appropriate Model

How does a user select a sound model for analysis? The first issue is that the sound model must be able to represent the input sound. For example, a sinusoids-in-a-harmonic-series model will clearly not be able to represent a very inharmonic sound (unless the model's fundamental frequency is extremely low). The second issue is that the sound model must support the desired modifications.

Rodet has the following recommendations: use the PSOLA model for voice, and for low pitch, harmonic sound where phase is important and where sinusoidal models need a phase model. (See "Interpolation of Sinusoidal Parameters" below for more on phase models.) LPC allows

spectral transformations that time-domain PSOLA does not. LPC and resonance models can be used to filter other sounds.

Fitz and Haken claim that the strengths and weaknesses of various models are best evaluated in the context of particular tasks, rather than particular sounds. One of the strengths of Loris' model is that it yields high-fidelity representations of a great variety of sounds (cats, drums, apples, trombones, etc) without sacrificing the homogeneity of the representation and, therefore, the continuous transformability of the model data. Because this representation contains only one type of component, it can be used to perform continuous, smooth morphs between sounds of very different character. For manipulations and transformations in which this homogeneity of data is important, this model has a significant advantage.

Serra argues that the selection of the appropriate model depends not only on the sound used and the specific task, but also on compromises to be defined by the user such as sound fidelity, flexibility, generality, memory consumption and compute time.

In addition to these factors above, Wakefield also notes that the sensitivity of each method to violations in the signal assumptions needs to be factored into any decision regarding choice of model. Methods that implicitly require knowledge of the source's pitch can be susceptible to those measurement factors that corrupt the pitch estimate. The presence of reverberation, distortion, or multiple sources in the original recordings may also affect the accuracy of the parameters extracted and the quality of the resulting synthesis.

Choosing an Inappropriate Model

Sometimes it can be musically useful to represent a sound with a model that, in principle, is inappropriate for that sound.

For example, while it may seem that SNDAN's *pvan* method would only work for tones with harmonic partials, it actually works well in a variety of situations. This is because the filter bank does an excellent job of covering the audio frequency range, so that all frequencies are represented. Nevertheless, the method works best when no more than one sine wave in the original sound falls within each filter, and sine waves do not cross between bins. When sounds are expanded in time (with no pitch change), we note that, depending on the choice of analysis frequency, there are sometimes window-size-related modulation artifacts.

Similarly, IRCAM's resonance model analysis program *ModRes* (Potard, et al., 1986) was designed for percussive, impulse-like sounds, but musicians have used it with sustained sounds.

MDRx takes advantage of the sparse nature of the time-frequency surface to clean-up noise-corrupted signals. In this case, the measured sound consists of a proper signal with sparse surface properties and other signals. By fitting the sparse properties of the surface, it is possible to eliminate, in some cases, extraneous wideband sources.

The Deterministic plus Stochastic model that is part of SMS is not valid for all sounds; still many users force it to a particular sound as a way to get new effects. This is also the case when forcing a sinusoidal harmonic model into an inharmonic sound or when forcing the concept of very stable partials into a very fast changing sound.

Interpolation of Sinusoidal Parameters

All but one[2] of the analysis/synthesis systems in this comparison sample the parameters of the model (e.g., frequency and amplitude of sinusoids) at intervals in the range of 1-10 milliseconds. A simplistic additive synthesizer that held each sinusoids' parameters constant over this interval and then changed values abruptly at the next frame would produce very noticeable artifacts: a click or other discontinuity at each frame. Synthesizers must therefore smoothly interpolate parameters between frames, and in many cases the analysis must make assumptions about how the synthesizer will interpolate. How does each analysis/synthesis system handle this issue?

A related question concerns the relationship between frequency and phase. Many analysis techniques for sinusoidal models determine both a frequency and a phase for each sinusoid in each frame. Because frequency and phase are not independent quantities (frequency is the derivative of phase), the storage of both can be considered an overspecification. How should a synthesizer behave when a partial's stated phase at a given time differs from the phase computed by integrating the continuous frequency since the previous phase value?

CNMAT's *TASS* ("Trivial Additive Synthesizer for SDIF") starts each sinusoid with the correct initial phase, then ignores all subsequent phase values for that sinusoid, instead using the integral of instantaneous frequency. *TASS* linearly interpolates each partial's amplitude and frequency between frames.

SNDAN's resynthesis of *pvan* sound models works exactly the same way, starting with initial phases for the first frame, then linearly interpolating amplitudes and frequencies between frames. Another technique, to be released soon, employs quadratic phase interpolation to ensure correct phases at the frame points.

Loris also starts each sinusoid with the correct initial phase, then ignores all subsequent phase values for that sinusoid. (For Loris, the "start" of a sinusoid is any transition from zero amplitude to nonzero amplitude.)

An advantage of using only the initial phase for each sinusoid is that this technique easily supports time-scale and frequency modifications in the resynthesis.

SNDAN's resynthesis of *mqan* sound models uses cubic interpolation of phases between frames in order to preserve phase and frequency continuity at each frame point.

IRCAM's *additive* resynthesis currently uses third order polynomial interpolation for frequency, allowing it to synthesize each partial with both the frequency and the phase given at each frame's time. This interpolation scheme introduces artifacts, so IRCAM's analysis/synthesis team plans to change this as soon as possible. IRCAM's PSOLA resynthesis always preserves phase.

Amplitude is usually linearly interpolated in IRCAM software. However, Cepstrum and Discrete Cepstrum (Galas and Rodet, 1990) represent the envelope of the logarithm of the magnitude spectrum. Therefore, linearly interpolating cepstral coefficients results in logarithmic interpolating of amplitude. In the LPC domain, IRCAM's lattice-filter *filnor* can do reflection coefficients interpolation or interpolation of the so-called Log-Area-Ratio

[2] PartialBench instead provides a continuous parameter trajectory model based on splines. Spline breakpoints in PartialBench are typically more like 100 milliseconds apart.

coefficients[3] (which are easily derived from or transformed into reflection coefficients) (Markel and Gray, 1980).

PartialBench has been used to investigate the impact of the degree of the polynomials that represent the parameter trajectories. The effect on the analysis side has been outlined in the section on PartialBench above. In terms of spline functions the change of the interpolation technique can be interpreted as a change of the basis functions without changing weights of the basis functions. It can be argued that a change of the interpolation technique will introduce errors into the synthesized signal, however, these errors do not affect the main model characteristics: the number of partials and the bandwidth associated with each partial. The difference will in most cases have small audible consequences, however, it should be noted that differences in the phase interpolation may be more severe due to the nonlinearity, the cosine function, that is applied to the phase trajectory. While technically a change in the polynomial order is a change in the model with respect to the smoothness that can be obtained this change is generally of minor consequences for the use of the model and the mismatch between the interpolation expected during analysis and the interpolation taking place during synthesis is probably better termed a noise source than a change of the model.

[3] This latter interpolation scheme is better than reflection coefficients interpolation since it is like interpolating the underlying physical model, i.e., an acoustic tube made of a concatenation of sections with various areas. For example, in Log-Area-Ratio coefficient interpolation, formants appear to be interpolated in frequency and amplitude, while in reflection coefficient interpolation the corresponding spectral features appear to be cross-faded.

Röbel claims that a proper analysis algorithm should resolve any contradictions between phase and frequency. However, this is done under the assumption of a specific interpolation. If the synthesizer faces contradiction between frequency and phase values one can conclude that the interpolation chosen is different from the one that the analysis algorithm assumed.

Rodet claims that rather than storing instantaneous phase, it would be better to have phase models. A phase model is an algorithm that explicitly provides a phase value for each sinusoidal partial, even when transformations such as stretching or transposition are applied. Naturally, letting the phase run as the integral of instantaneous frequency is a sort of a phase model but a rather weak one since it depends in a complicated way on fine details of frequency tracks and frequency interpolation formulas. As an example, stretching a low-pitched voice with this model usually leads to rather unnatural voice sounds. On the contrary, a more sophisticated phase model might say that the phase of the partials should have a precisely defined value at each closure of the vocal folds (Peeters and Rodet, 1999), as found in the signal. With this model, such a voice can be stretched without loss of naturalness.

SMS has different possibilities for interpolating parameters from frame to frame. For example the amplitude can be interpolated linearly either in a dB scale or in a linear scale. Frequency and phase interpolation can also be done by simply interpolating frequencies starting with a fixed or random phase, by preserving the original phases using quadratic interpolation, or by using a phase model.

SDIF needs to be extended to represent interpolation method, so that if an analysis program makes a specific assumption about what interpolation will be used in resynthesis, that

assumption can be unambiguously represented in the SDIF file.

In spite of the differences among all these phase and interpolation models, in most cases the differences are not audible. Wright synthesized every participant's SDIF files with *TASS*, which uses the most simplistic possible models, and in most cases could not distinguish between the "correct" resynthesis and the *TASS* resynthesis. Further subjective comparisons are clearly needed; see "Future Work."

Modeling of Noise

What models are used to represent noise components (e.g., an apple bite or the breathiness of a shakuhachi) in these analysis/synthesis systems? What determines which parts of the signal will be treated as noise? How mutable is each noise model?

Neither of SNDAN's sound models separates the sinusoidal parts from the "residual." The noise residual is considered to be a noise modulation on the sinusoidal components, so they are represented by the microvariations of the amplitudes and frequencies of these components. The *mqan* model tends to model noise with many short-lived partials; this tends to produce artifacts when changing the time scale.

SMS and both of IRCAM's analysis models provide an explicit residual sound by subtracting the sinusoidal resynthesis from the original sound. In both cases the residual can be kept as a (non-mutable) sound file or modeled spectrally. IRCAM provides white noise filtered by spectral envelopes represented as discrete Cepstrum, LPC, Magnitude FFT, channels, etc. SMS approximates the magnitude spectrum of each frame by a linear interpolation using a given number of coefficients.

IRCAM's analysis software considers each magnitude peak of the Short-Time Fourier Transform ("STFT") as a potential sinusoid. The first test of "sinusoidalness" is how well the shape of the peak correlates with the shape of a pure sinusoid; a low correlation means that the peak will be rejected as a sinusoid. In *hmm* analysis there is a second test: a sinusoid is defined as the presence of energy in several (enough) successive frames with slowly (enough) varying frequency and amplitude. Parameters of the *hmm* analysis determine the allowable slope of a sinusoid's frequency and amplitude, and also the minimum length of time that a partial may be alive.

In IRCAM's analysis, any peak that is not considered a sinusoid will simply be discarded from the sinusoidal portion of the model. Therefore whatever energy appears at that point in the spectrum at that time will appear in the residual file and will therefore be considered noise.

Loris' bandwidth-enhancement represents the noisy components as part of each partial, as described in the section on Loris above.

With MDRx, the time/frequency surface represents all parts of the spectrum, including the noise, which are concentrated around ridges. Accordingly, we have found that in-band noise components, such as breathiness in the singing voice, "burples" in brass instruments, or the texture of the shakuhachi, are well-represented by the instantaneous amplitude and frequency contours extracted from the surface and yield in accurate synthesis. Wide-band noise components, such as fricatives in sung passages or bowing noise in closely-miked recordings of string instruments, are not well represented under MDRx, and clearly result in artifact (or absence) in the synthesized signal.

Mutability versus Accurate Resynthesis

Time-domain samples can be considered a perfectly accurate sound model (within the limits of human hearing and of playback through loudspeakers) with very little mutability. Within the additive synthesis domain, it is possible to sacrifice mutability completely to achieve perfect resynthesis. For example, one can always perfectly reconstruct *n* samples of a waveform using *n* fixed-frequency sinusoids, at the expense of being able to apply any kind of manipulation in general. At the opposite extreme, a fundamental frequency trajectory is a sound model that can be resynthesized by applying it to a synthetic timbre such as a sawtooth wave. This sound model is completely mutable, but does not allow for accurate resynthesis at all.

Is there a tradeoff between mutability and accurate resynthesis among different sound models?

As described in the previous section, the *mqan* model is not very mutable when there are lots of short-lived partials.

Harmonic series tend to be very mutable, as do noise models based on a specified time-varying spectral envelope.

Serra argues that there is a tradeoff. For example an STFT resynthesis can result into a perfect reconstruction but does not permit many of the transformations that another model, such as a harmonic sinusoidal model, would give at the expense of less accuracy of resynthesis.

Fitz and Haken maintain that in general the optimal representation for a particular sound will maximize mutability and fidelity. However, they do not feel that they have perfected their modeling techniques, and sometimes are unable to achieve such a representation. So the representation is sometimes compromised for suitability for a particular task.

Rodet claims there is no tradeoff: The more accurate the analysis (hence the synthesis) the more mutable it is, or said in another way, for a given analysis-synthesis method, improving accuracy (according to the model) will not degrade mutability. If there is such a tradeoff between mutability and accuracy, it depends of the choice of the analysis method (FFT, LPC, additive, etc.).

Here is Röbel's view: With STFT based techniques there exist a infinite number (one for each FFT size) of perfect signal representations. In fact a time-series representation can be viewed as a STFT with window/ FFT size one. Depending on the signal characteristics some can be used for modifications while most of them can not. STFT is closely related to a fixed frequency and fixed amplitude additive model. Extending the model by allowing amplitude modulations one can argue that this can be related on the analysis side to wavelet techniques, which already provide a multitude of infinite sets of exact representations (namely, one for each class of wavelets and type of partitioning of the time frequency plane). A general additive model with time varying amplitude and time varying frequency is a superset of all these models and, therefore, provides an even larger space of exact representations. For example it can be shown that you can exactly substitute a single amplitude/frequency modulated harmonic by a collection of *n* amplitude/frequency modulated harmonics. Given the experience with the phase vocoder it is clear that only a small portion of these are mutable models. As Fitz and Haken stated, the main problem is to locate the exact representation model with the proper mutability properties.

Modeling the Gong sound with PartialBench provided an interesting example. Röbel tried

successfully to separate two resonances that are quite close in frequency and result in a slow beating of the sound. Time-stretching the sound based on this model fairly reproduced the beating. For transposition however, this model will change the beating and a single amplitude-modulated partial, which can be derived from the two independent partials, would have to be used to preserve the sound quality. This example demonstrates that the selection of the proper model is related to the task to be performed and is a simple example for the huge space of perfectly reconstructing models and the differing mutability properties.

Mutability – Time and Pitch Modification

SNDAN comes with several synthesizers that offer different forms of time and pitch dilation on resynthesis, allowing, for example, different stretch factors to be applied to the attack, steady state, and decay portions of single sounds. Note that for single-pitched harmonic sounds, if the analysis frequency for *pvan* is chosen correctly (i.e., equal to the pitch frequency of the sound), the window size is optimum, and mutability is excellent. If the sound is grossly inharmonic or if the frequencies change by more than a few percent, a compromise analysis frequency (window size) must be found.

Loris supports time- and frequency-scale dilation.

SMS offers different types of time stretching and pitch scaling methods.

IRCAM's *additive* model with noise and transient modeling is able to retain sharp attacks when modifying a sound's time or pitch scale. IRCAM's spectral envelope modeling allows for natural-sounding transposition of voice without altering formant structures. IRCAM's PSOLA modeling also provides satisfactory time-stretching of voice examples.

Mutability – Morphing

Timbral interpolation (or "morphing") interpolates the parameters of two or more sound models to produce an intermediate sound sharing characteristics of the two input sounds. Typically this interpolation is a weighted average, providing continuous control of the output sound's position between two input sounds, or, more generally, of the output sound's position in a timbre space (Wessel, 1979).

Morphing is an art form; there is no single "correct" morph among a given set of sounds. The success of a morph depends on a number of perceptual factors. In general if the input sounds share a perceptual feature (e.g., "a single pitched note"), the morph result should also share that feature. Any perceptual parameters that differ among the input sounds (e.g., brightness) should change smoothly as the interpolation weighting changes.

Additive synthesis supports morphing at a low level because each partial's frequency and amplitude parameters can be a weighted average of the frequencies and amplitudes of the corresponding partials in the input sounds. Morphing among harmonic models is straightforward because the notion of "corresponding" partials can come from harmonic number: the parameters of nth harmonic partial of the output come from the parameters of the nth harmonic partials of the inputs. Morphing among more general additive models is more difficult, because there is no obvious way to find "corresponding" partials.

Time-scale issues must also be addressed: what should be the result when morphing a 1-second sound with a 2-second sound? More challenging time-scale morphing difficulties include sounds with different number of vibrato cycles, and

sounds with and without release segments. Sounds that are not monophonic instrument tones and that have different numbers of temporal features to align are even more challenging (Tellman, et al., 1995).

Now that all of these analysis systems output SDIF, it would be possible to create models with one (or more) analysis programs, and morph the results together in the SDIF domain. For now, however, morphing capabilities are built into specific systems rather than embodied in a general way as a program that operates on SDIF data.

IRCAM's *interpolator* program (http://www.ircam.fr/anasyn/perry/interpolator) supports interpolation between harmonic series (or pairs of harmonic series in the case of the "run *additive* twice" model) with individual control of the interpolation weights for amplitudes, frequencies, and spectral shape of the noise.

Loris's ressigned bandwidth-enhanced model excels at morphing. Sound morphing is achieved by interpolating the time-varying frequencies, amplitudes, and bandwidths of corresponding partials obtained from reassigned bandwidth-enhanced analysis of the source sounds. Loris establishes correspondences between partials by the process called "distillation." The frequency spectrum is partitioned into channels having time-varying widths and center frequencies, and each channel having a unique identifier. All partials falling in a particular channel are distilled into a single partial, which is assigned the corresponding identifier, leaving at most a single partial per frequency channel. Channels that contain no partials are not represented in the distilled partial data. Distilled partials having the same identifier are morphed by interpolating their frequency, amplitude, and bandwidth envelopes according to a specified morphing function.

The various morph sources need not be distilled using identical sets of frequency channels. Distilled partials in one source that have no corresponding partial in the other source(s) are faded in and out according to the morphing function. If most of the source partials have no corresponding partial in the other source(s), then the dominant effect will be a crossfade, rather than a morph. Moreover, dramatic partial frequency sweeps will dominate other audible effects of the morph, so care must be taken to coordinate the frequency channels used in the distillation process.

Significant temporal features of the source sounds, such as the beginning and end of the sound, or the attack and decay portions of instrument tones, must be synchronized in order to achieve good morphing results. Typically, when synchronizing a sound morph with, for example, a visual animation, temporal features of the source sounds are aligned with visual events in the animation. The process of synchronizing temporal features by expanding or compressing the time scale of the source partials is called time dilation. Time dilation can occur before or after distillation, but is an essential component in controlling the temporal evolution of the morph.

SMS supports morphing not only in the sinusoidal parameter domain, but also morphing of the higher-level parameters such as fundamental frequency, spectral shape, and spectral tilt that SMS analysis can extract.

SNDAN does not provide morphing.

MDRx has been used to study morphing based on models other than time-varying partials. Our most extensive work has concerned singer classification. We have studied the perceptual consequences of morphing the AR model of the composite transfer function within one register across singers or across registers within a singer. These generally provide a continuous variation

in the perceived attributes of the synthesized voice. We have also manipulated the pole clusters in piano synthesis to achieve interesting effects in the attack transient. Finally, by adjusting the proximity of the poles and zeros to the unit circle, we have adjusted the degree of amplitude modulation present in the vibrato violin signal to create violins of different acoustic characteristics.

Mutability – Other Modifications

SNDAN supports various modifications of harmonic sound models. For example, amplitude-versus-time and frequency-versus-time functions can be smoothed and altered in various ways. Also, two models can be combined. For example, the harmonic amplitudes of one sound can be scaled in order to achieve the spectrum of another sound at some point in time with the result that the resynthesized sound has the time-varying character of one sound and the overall spectral character of the other. Some types of modifications may result in substantial reductions in the amount of data necessary for resynthesis without introducing much degradation in quality.

Loris' model accommodates a variety of model-domain transformations, including compression and warping, spectral envelope shaping (filtering), etc.

Choosing Analysis Parameters

Most analysis systems provide many parameters that the user must set in order to achieve good results. What parameters must a user (such as a composer) set to run each analysis tool, and how can a user choose effective settings for these parameters?

MDRx requires a single signal-dependent parameter for extracting the time-frequency surface of the signal: the minimum frequency separation between partials. The value of this parameter automatically determines the length of the signal used to calculate the frequency slice of the surface at each point in time. In practice, FFT's are used to speed up the computation of the MDRx so that signal lengths range from 1k-8k for frequency for separations between 50 and 800 Hz at a sampling rate of 44.1 kHz. This parameter does not have to be precisely known. In our studies, we have observed that the results are insensitive to plus or minus a half octave of the actual frequency separation. Conversion from a Modal Distribution representation to sinusoidal or other models requires the user to set additional parameters.

Analysis for harmonic additive models generally begins with a fundamental frequency estimate. With SNDAN's *pvan* the user must supply a single fundamental frequency value for the entire file. SMS and IRCAM's *additive* analysis begin with an fundamental frequency estimation pass that produces a fundamental frequency envelope; both systems allow the user to constrain the allowable values of the fundamental frequency, which can help prevent octave errors.

Fourier-based analysis methods generally include a window size parameter. A central idea of SNDAN's *pvan* is that the window size is exactly twice the pitch period; this way the bin frequencies exactly line up with the frequencies of the (presumed) harmonic partials and there is no need for any further processing to connect partials over time into tracks. SMS and IRCAM's *additive* pick a default window size based on the fundamental frequency, supporting heuristics such as "the window size should include 3-5 pitch periods." Even MDRx, though it is not Fourier-based, has a window size parameter.

IRCAM's *additive* analysis has parameters for the number of partials to look for and the width of the "harmonic sieve" used to find each partial.

SMS analysis has many of the same parameters, but provides default values for these parameters depending on a user-selected "sound type," e.g., "Sound Phrase." SMS comes with a detailed software manual (http://www.iua.upf.es/sms/dist/docs/sms_manual_html/book1.htm) that provides a lot of information on the analysis parameters and how to set them according to the attributes of the sound being analyzed.

Rodet and the SMS manual each recommend looking at a graph of the sound spectrum to help select analysis parameters.

SNDAN's *mqan* method works very well for a variety of sounds, and it works especially well for sounds whose frequencies vary over considerable ranges. The two big factors that limit its robustness are the "spectral threshold" and the tracking algorithm. While the user sets the first, the second is very complex and is hard-wired into the program. Imperfect tracking causes spectral artifacts that are especially audible in the higher register, particularly if the file is resynthesized with an expanded time scale. The spectral threshold effects the degree to which noise-like spectral components are dropped out. While keeping a certain amount of these is necessary for naturalness, keeping too many can cause tracking problems. Another factor that is varied by the user is the "minimum frequency of analysis." This directly affects window size. On some sounds, like bongo hard hits, the user can trade off transient (impact) accuracy against spectral accuracy. It may not be possible to find a value that results in a perfect replica of the input signal, since either a distorted transient or a distorted spectrum will occur in varying amounts.

Loris has a handful of parameters that can be adjusted to optimize the representation of a particular sound. The developers have made a sustained effort to make those parameters orthogonal and hierarchical. "Orthogonal" means here that optimal settings of one parameter are not a function of the settings of any other parameter, and interaction between the various parameters is minimal. "Hierarchical" means here that a single parameter (frequency resolution: minimum instantaneous frequency separation between partials) can be used to configure the analysis and to quickly obtain an informative, if not high-fidelity, representation. If necessary, another parameter (usually the width of the analysis window in the frequency domain) can be adjusted to improve the representation, and so on. In general, the useful region of the overall parameter space can be found using a single intuitive parameter. Users rarely need to set more than three parameters to obtain good results.

For PartialBench there are three parameters: First the order of the polynomial used for parameter trajectories. This is generally not necessary to adjust and is currently fixed to third order polynomials for amplitude and phase. Second the number of partials, which is simply increased until the model quality is sufficient. The third and most critical parameter determines the node positions of the spline basis functions. This parameter is equivalent to the window size parameter for traditional analysis schemes. It defines the frequency resolution of the analysis and the bandwidth of the individual partials. Still there exists no straightforward method to adjust it and one usually should have a look onto the time signal and the frequency spectrum to find proper values.

Artifacts of Resynthesis

What are the characteristic "artifacts" of various techniques, and how can a user learn to adjust the analysis parameters when she hears these artifacts?

A few brief attempts to characterize these sounds in words: Incorrect births and deaths of partials

can result in short whistles. Attempting to represent noisy spectral components with very short-lived sinusoids can result in a sort of "buzziness." The harmonic sieve process of IRCAM's *additive* analysis will be ruined by fundamental frequency errors, resulting in very fast glissandi of all partials and "bloups and cracks" type artifacts.

There are some characteristic artifacts of certain classes of suboptimal parameter configurations in Loris. The most common problem is probably insufficient frequency resolution. This causes partials near to each other in frequency to cut in and out when their frequency difference is near the minimum allowed difference. In very bad cases, this causes a "crunching" effect or sometimes other, less objectionable artifacts. This can be observed with the harp glissando and unison soprano voice examples, because in both cases many sustaining harmonic partials overlap temporally. Other less dramatic artifacts related to this problem include reverb suppression, for the same reasons. Poor choices of analysis window cause these effects too, and also may have the effect of smearing temporal features, when the window is too long. With experience, a user can often recognize and compensate for these artifacts.

Summary of Criteria for Evaluating Sound Models

Although a definitive comparison of the methods eluded the panel, an important result of this first coordinated effort was to establish the groundwork for comparative evaluations.

Here are the criteria we found for evaluating a sound model for analysis/synthesis:

- Does it capture all the features of a given input sound?
- What artifacts are introduced?
- Does it capture features in an isolated way so they can be manipulated? Manipulations include the following:
  - Exaggerate/reduce features
  - Combine features from separate models
  - Interpolate/extrapolate
- Coding efficiency (data compression)
- Capture efficiency / coder cost
- Ability to make tradeoffs between efficiency and feature capture (Chaudhary, Projected completion in May, 2001)

Conclusions and Future Work

We have compared six of the main sound analysis/synthesis systems for music, detailing differences in the sound models used by each system. Each of these systems now outputs SDIF files, which made this comparison possible and facilitates future comparison and interchange among these systems. Although we did not systematically compare all six systems' analysis results for each of the 27 input sounds, all analysis results are available online at http://www.cnmat.berkeley.edu/SDIF/ICMC2000

There is no conclusion as to which is the best analysis/synthesis system. We have provided some criteria for selecting an appropriate analysis/synthesis system based on the sound model used by the system, the availability of that system on the desired platform(s), the kinds of sounds that can be modeled with each system, and the desired kinds of transformations of the analyzed sound.

Further comparison of the analysis result SDIF files is clearly warranted as future work, and we encourage others to download and study the SDIF files, and to share their results with the community.

We would like to develop objective measures (where possible) of the criteria for evaluating sound models.

More SDIF integration is also needed. Although each system's analysis component now outputs SDIF, not all system's synthesis components can read SDIF. SNDAN's *mqan* model to *pvan* model converter could be made into a useful SDIF "1TRC" model to "1HRM" model converter. IRCAM's *interpolate* program would also be useful as a general-purpose SDIF tool.

SDIF needs a representation for interpolation method, as described above.

There are many useful ways to incorporate the SDIF Stream Relationships Language (Wright, et al., 2000) into the SDIF output of these six systems.

Acknowledgements

Richard Andrews, Peter Castine, Amar Chaudhary, Adrian Freed, Leigh Landy, Timothy Madden, Diemo Schwarz, Paula Spiese, Patrice Tisserand, David Wessel, Michael Zbyszynski.

Sound Examples

The first 27 sound examples are the original input sounds:

1. Angry cat (5.78 sec)
2. Trumpet note (2.43 sec)
3. Abbie Conant flutter-tonguing a trombone (2.39 sec)
4. Saxophone phrase with interesting articulation (3.97 sec)
5. Same clarinet phrase as "deplus", no reverb (1.85 sec)
6. Musical phrase played on a suling flute (7.86 sec)
7. Shakuhachi phrase (13 sec)
8. Clarinet phrase "deplus", lots of reverb (1.79 sec)
9. Art Tatum piano excerpt (monophonic line) (2.69 sec)
10. Acoustic guitar playing a monophonic line (2.55 sec)
11. Biting into an apple (0.9 sec)
12. Berimbao rhythm (3.5 sec)
13. Bongo roll (1.25 sec)
14. Xylophone note (0.781 sec)
15. Low bass drum note (2.52 sec)
16. Piano note (3.13 sec)
17. Low gamelan gong (with fade-out) (5.65 sec)
18. Harp glissando (5.13 sec)
19. Chorus singing Webern (5.33 sec)
20. Snippet of a tampura drone (0.994 sec)
21. Shafqat Ali Khan singing a phrase from Raga Derbari (1.63 sec)
22. Singing into and playing a flute at the same time (1.63 sec)
23. Yodelling (1.47 sec)
24. Sopranos singing in unison (1.87 sec)
25. Giggle (0.971 sec)
26. Shafqat Ali Khan saying "research" (0.695 sec)
27. 10 simultaneous chirps (0.21 sec)

The next 10 sound examples are resynthesized modified sound models from IRCAM.

28. Angry cat PSOLA model time-stretched by a factor of 2 (11.57 sec)
29. "Research" speech PSOLA model time-stretched by a factor of 4 (2.79 sec)
30. Trombone PSOLA model time-stretched 2x and drastic change of pitch envelope (4.79 sec)
31. Trumpet note PSOLA model time-stretched 2x and pitch-shifted down drastically (4.84 sec)

32. Xylophone resonance model resynthesis (2.54 sec)
33. Bongo roll sinusoidal model without transients, synthesized (1.26 sec)
34. Bongo roll sinusoidal model with transients, synthesized (1.26 sec)
35. Apple bite filtered noise and transient model, synthesized (0.89 sec)

The remaining sound examples are resynthesized modified sound models and morphs from Loris:

36. Angry cat time-compressed (2.48 sec)
37. Saxophone phrase time-stretched by a factor of 2 (8.01 sec)
38. Apple bite/bass drum morph (3.09 sec)
39. Morph of angry cat frequencies and trombone amplitudes (2.48 sec)
40. Morph from angry cat to trombone (2.48 sec)
41. Long trombone/cat morph (9.07 sec)
42. Morph from trombone to angry cat (2.48 sec)
43. Piano/Xylophone morph step 1 (3.23 sec)
44. Piano/Xylophone morph step 2 (3.23 sec)
45. Piano/Xylophone morph step 3 (3.23 sec)
46. Piano/Xylophone morph step 4 (3.23 sec)
47. Piano/Xylophone morph step 5 (3.23 sec)
48. Piano/Xylophone morph step 6 (3.23 sec)
49. Piano/Xylophone morph step 7 (3.23 sec)
50. Piano/Xylophone morph step 8 (3.46 sec)
51. Piano/Xylophone morph step 9 (3.46 sec)
52. Piano/Xylophone morph step 10 (3.46 sec)

References

Auger, F. and P. Flandrin 1995. Improving the Readability of Time Frequency and Time Scale Representations by the Reassignment Method. *IEEE Transactions on Signal Processing*, 43, 1068-1089.

Beauchamp, J. W. 1993. Unix Workstation Software for Analysis, Graphics, Modification, and Synthesis of Musical Sounds. *Proceedings of the Audio Engineering Society*, vol. Preprint #3479.

Chaudhary, A. (Projected completion in May, 2001). Perceptual Scheduling in Real-Time Music and Audio Applications. Unpublished PhD Dissertation, Electrical Engineering and Computer Science, University of California, Berkeley, CA.

Cohen, L. 1995. *Time-frequency analysis*. Englewood Cliffs, New Jersey: Prentice-Hall.

Crochiere, R. E. 1980. A weighted overlap-add method of short-time Fourier analysis/synthesis. *IEEE Transactions on Acoustics, Speech and Signal Processing*, ASSP-28(1), 99-102.

Depalle, P., G. Garcia, and X. Rodet 1993. Tracking of partials for additive sound synthesis using hidden Markov models. *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Minneapolis, MN, USA, pp. 225-8. (http://www.ircam.fr/equipes/analyse-synthese/listePublications/articlesRodet/ICASSP93HMM)

Dolson, M. 1986. The Phase Vocoder: A Tutorial. *Computer Music Journal*, 10(4), 14 - 27.

Doval, B. and X. Rodet 1993. Fundamental frequency estimation and tracking using maximum likelihood harmonic matching and HMMs. *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 221-4 vol.1.

Fitz, K. and L. Haken 1995. Bandwidth Enhanced Sinusoidal Modeling in Lemur. *Proceedings of the ICMC*, Banff, Canada.

Fitz, K., L. Haken, and P. Christensen 2000a. A New Algorithm for Bandwidth Association in Bandwidth-Enhanced Additive Sound Modeling. *Proceedings of the ICMC*, Berlin, Germany, pp. 384-387.

Fitz, K., L. Haken, and P. Christensen 2000b. Transient Preservation Under Transformation in an Additive Sound Model. *Proceedings of the ICMC*, Berlin, Germany, pp. 392-395.

Freed, A. and T. Jehan 1997. Database of Challenging Musical Sounds for Evaluation and Refinement of Pitch Estimators. *Proceedings of the ICMC*, Thessaloniki, Hellas. (http://cnmat.CNMAT.Berkeley.EDU/ICMC97/papers-html/Pitch.html)

Galas, T. and X. Rodet 1990. An improved Cepstral Method for Deconvolution of Source Filter Systems with Discrete Spectra: Application to Musical Sound Signals. *Proceedings of the ICMC*, Glasgow, pp. 82-84.

Guevara, R. and G. H. Wakefield 1996. A Modal Distribution Approach to Piano Analysis and Synthesis. *Proceedings of the Intl. Comp. Music Conf.*, Hong Kong.

Guevara, R. C. L. (1997). Modal Distribution Analysis and Sum of Sinusoids Synthesis of Piano Tones. Electrical Engineering and Computer Science, University of Michigan, Ann Arbor.

Helmholtz, H. von1875. *On the sensations of tone as a physiological basis for the theory of music*. New York: Dover Publications.

Kronland-Martinet, R. 1988. The Wavelet Transform for Analysis, Synthesis, and Processing of Speech and Music Sounds. *Computer Music Journal*, 12(4), 11 - 20.

LDC 2000. Linguistic Data Consortium home page. http://www.ldc.upenn.edu

Madden, T. and J. Beauchamp 1999. Real Time and Non-Real Time Analysis of Musical Sounds on a Power Macintosh. *Proceedings of the ICMC*, Beijing, China, pp. 411-413.

Maher, R. C. 1990. Evaluation of a method for separating digitized duet signals. *AES*, 38(2-3), 219-245.

Maher, R. C. and J. W. Beauchamp 1994. Fundamental Frequency Estimation of Musical Signals Using a Two-Way Mismatch Procedure. *Journal of the Acoustical Society of America*, 95(4), 2254-63.

Markel, J. D. and A. H. Gray, Jr 1980. *Linear Prediction of Speech*: Springer.

McAulay, R. J. and T. F. Quatieri 1985. Mid-rate coding based on a sinusoidal representation of speech. *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, Tampa, FL, USA, pp. 945-8.

McAulay, R. J. and T. F. Quatieri 1986. Speech Analysis/Synthesis Based on a Sinusoidal Representation. *IEEE Trans. Acoustics, Speech, and Signal Processing*, ASSP-34(4), 744-754.

Mellody, M. (2000). Signal analysis of the female singing voice: features for perceptual singer identity. Ph.D. dissertation, Applied Physics Program, The University of Michigan, Ann Arbor.

Mellody, M. and G. H. Wakefield 2000a. Signal Analysis of the Singing Voice: Low-Order Representations of Singer Identity. *Proceedings of the ICMC*, Berlin, pp. 98-101. (http://www.eecs.umich.edu/~ghw/Papers/icmc00_singer_slides_final.pdf)

Mellody, M. and G. H. Wakefield 2000b. The Time-Frequency Characteristics of Violin Vibrato: Modal Distribution Analysis and

Synthesis. *Journal of the Acoustical Society of America*, 107(1), 598-611.

Mellody, M., G. H. Wakefield, and F. Herseth 2000. Modal Distribution Analysis and Synthesis of a Soprano's Sung Vowels. *Journal of Voice*, submitted for publication.

Moorer, J. A. 1978. The use of the phase vocoder in computer music applications. *Journal of the Audio Engineering Society*, 26(1-2), 42-5.

Mrozek, E. 1999. "Adaptive time-frequency representation for trumpet sounds and other nearly harmonic signals," University of Michigan, Ann Arbor, Doctoral research summary and thesis proposal.

Peeters, G. and X. Rodet 1998. Signal Characterization in terms of Sinusoidal and Non-Sinusoidal Components. *Proceedings of the DAFX*, Barcelona, Spain.

Peeters, G. and X. Rodet 1999. SINOLA: A New Analysis/Synthesis Method using Spectrum Peak Shape Distortion, Phase and Reassigned Spectrum. *Proceedings of the ICMC*, Beijing, China, pp. 153-156.

Pielemeier, W. J. and G. H. Wakefield 1996. A high resolution time-frequency representation for musical instrument signals. *Journal of the Acoustical Society of America*, 99(4), 2382-2396.

Pielemeier, W. P. (1993). Representation and Estimation Methods for Transcription of Musical Signals. Ph.D., Electrical Engineering and Computer Science, The University of Michigan, Ann Arbor.

Portnoff, M. R. 1976. Implementation of the digital phase vocoder using the fast Fourier transform. *IEEE Transactions on Acoustics, Speech and Signal Processing*, ASSP-24(3), 243-8.

Potard, Y., P.-F. Baisnée, and J.-B. Barriere 1986. Experimenting with Models of Resonance Produced by a New Technique for the Analysis of Impulsive Sounds. *Proceedings of the ICMC*, La Haye, pp. 269-274.

Risset, J.-C. and D. Wessel 1999. Exploration of Timbre by Analysis Synthesis. In Deutsch, D. (ed.) *The Psychology of Music*, Second edition. San Diego: Academic Press.

Risset, J. C. and M. V. Mathews 1969. Analysis of musical-instrument tones. *Physics Today*, 22(2), 23-32.

Röbel, A. 1999. Adaptive Additive Synthesis of Sound. *Proceedings of the ICMC*, Beijing, China, pp. 256-259. (http://www.kgw.tu-berlin.de/~roebel/paper/icmc99.ps.gz)

Rodet, X. 1997a. The Diphone program: New features, new synthesis methods, and experience of musical use. *Proceedings of the ICMC*, Thessaloniki, Hellas, pp. 418-421.

Rodet, X. 1997b. Musical sound signal analysis/synthesis: sinusoidal-plus-residual and elementary waveform models. *Applied Signal Processing*, 4(3), 131-41.

Rodet, X. 2000. Sound analysis, processing and synthesis tools for music research and production. *Proceedings of the XIII CIM00*, l'Aquila, Italy.

Rodet, X., P. Depalle, and G. Poirot 1987. Speech Analysis and Synthesis Methods Based on Spectral Envelopes and Voiced/Unvoiced Functions. *Proceedings of the European Conference on Speech Tech.*, Edinburgh, U.K.

Rodet, X., Y. Potard, and J.-B. Barrière 1984. The CHANT Project: From Synthesis of the Singing Voice to Synthesis in General. *Computer Music Journal*, 8(3), 15-31.

Schafer, R. W. and L. R. Rabiner 1973. Design and simulation of a speech analysis-synthesis system based on short-time Fourier analysis. *IEEE Transactions on Audio and Electroacoustics*, AU-21(3), 165-74.

Schwarz, D. and X. Rodet 1999. Spectral Envelope Estimation and Representation for Sound Analysis-Synthesis. *Proceedings of the ICMC*, Beijing, China, pp. 351-354.

Schwarz, D. and M. Wright 2000. Extensions and Applications of the SDIF Sound Description Interchange Format. *Proceedings of the ICMC*, Berlin, pp. 481-484.

Serra, X. (1989). A System for Sound Analysis/Transformation/Synthesis Based on a Deterministic Plus Stochastic Decomposition. PhD dissertation, STAN-M-58, Music, CCRMA, Stanford University.

Serra, X. and J. Smith, III 1990. Spectral modeling synthesis: a sound analysis/synthesis system based on a deterministic plus stochastic decomposition. *Computer Music Journal*, 14(4), 12-24.

Smith, J. O., III and X. Serra 1987. PARLSHL: An Analysis/Synthesis Program for Nonharmonic Sounds Based on a Sinusoidal Representation. *Proceedings of the ICMC*, Champaign/Urbana, pp. 290-297.

Sterian, A. (1999). Model-based segmentation of time-frequency images for musical transcription. Ph.D., Department of Electrical Engineering and Computer Science, The University of Michigan, Ann Arbor.

Sterian, A., M. Simoni, and G. H. Wakefield 1999. Model-Based Musical Transcription. *Proceedings of the Intl. Comp. Music Conf.*, Beijing.

Sterian, A. and G. H. Wakefield 1998. Automatic Music Transcription: Modal TFD Analysis, State Estimation, and Kalman Filtering. *Proceedings of the Intl. Symp. on Opt. Sci., Eng., and Instr., SPIE'98*, San Diego.

Tellman, E., L. Haken, and B. Holloway 1995. Timbre morphing of sounds with unequal numbers of features. *Journal of the Audio Engineering Society*, 43(9), 678-89.

Wakefield, G. H. 2000. A Mathematical/Psychometric Framework for Comparing the Perceptual Response to Different Analysis-synthesis Techniques: Ground Rules For A Synthesis Bake-off. *Proceedings of the ICMC*, Berlin, pp. 229-232. (http://www.eecs.umich.edu/~ghw/Papers/icmc00_PerceptualComparisons.PDF)

Wakefield, G. H., M. Mellody, and N. D. Hogikyan 2000. Acoustic modeling of spasmodic dysphonia. *J. Voice*, submitted.

Wessel, D. L. 1979. Timbre space as a musical control structure. *Computer Music Journal*, 3(2), 45-52.

Wright, M., A. Chaudhary, A. Freed, S. Khoury, and D. Wessel 1999. Audio Applications of the Sound Description Interchange Format Standard. *Proc. of the Audio Engineering Society 107th Convention*. (http://cnmat.CNMAT.Berkeley.EDU/AES99/docs/AES99-SDIF.pdf)

Wright, M., A. Chaudhary, A. Freed, S. Khoury, and D. Wessel 2000. An XML-based SDIF Stream Relationships Language. *Proceedings of the International Computer Music Conference*, Berlin, Germany.

Wright, M., A. Chaudhary, A. Freed, D. Wessel, X. Rodet, D. Virolle, R. Woehrmann, and X. Serra 1998. New Applications of the Sound Description Interchange Format. *Proceedings of*

the International Computer Music Conference, Ann Arbor, Michigan, pp. 276-279. (http://cnmat.CNMAT.Berkeley.EDU/ICMC98/papers-pdf/SDIF.pdf)