COMPRESSION OF SINUSOIDAL MODELING PARAMETERS

Sylvain Marchand

SCRIME - LaBRI, Université Bordeaux 1
351, cours de la Libération, F-33405 Talence cedex, France
sm@labri.u-bordeaux.fr

ABSTRACT

In this paper we present a technique for lossless compression of the sinusoidal modeling parameters. Compression is indeed useful for embedding spectral sounds in a synthesizer, broadcasting spectral sounds or simply storing many of them on a medium.

This technique consists in compressing the frequency and amplitude parameters of each partial by adaptively sub-sampling and encoding their evolutions with time. It can be easily extended to handle spectral envelopes instead of partials, that is two-dimensional structures instead of one-dimensional ones.

We have also designed a very compact file format for (compressed) spectral sounds based on this compression scheme.

1. INTRODUCTION

Spectral models provide general representations of sound in which many audio effects can be performed in a very natural and musically expressive way. Based on additive synthesis, they contain a deterministic part consisting of a – often huge – number of partials, which are pseudo-sinusoidal tracks for which frequencies and amplitudes evolve slowly with time. The spectral modeling parameters of this deterministic part consist of the evolutions in time of the controls of the partials, thus leading to a large amount of data.

In this paper we present a technique for lossless compression of these sinusoidal modeling parameters. This technique can turn out to be very useful for analysis / synthesis programs dealing with spectral modeling. Section 2 clarifies the spectral model considered here, while Section 3 explains the compression technique for the model parameters. Finally, Section 4 presents some interesting improvements that can be made to the compression method.

2. SINUSOIDAL MODELING

The phase vocoder [1] uses the short-time Fourier transform to produce series of short-term spectra taken on successive windowed temporal frames. The McAulay-Quatieri analysis [2] looks across these short-term spectra for partials. These partials are pseudo-sinusoidal tracks for which frequencies and amplitudes vary slowly over time. The audio signal a can be calculated from the additive parameters using equations:

$$a(t) = \sum_{p=1}^{P} a_p(t)\, cos(\phi_p(t)) \quad (1)$$

$$\phi_p(t) = \phi_p(0) + 2\pi \int_0^t f_p(u)\, du \quad (2)$$

where P is the number of partials and the functions f_p, a_p, and ϕ_p are the instantaneous frequency, amplitude, and phase of the p-th partial, respectively. The P pairs (f_p, a_p) are the parameters of the sinusoidal model and represent at time t points in the frequency-amplitude plane. This representation is used in many analysis / synthesis programs such as Lemur [3], SMS [4, 5], S+T+N [6, 7], as well as our *InSpect* [8, 9] free software package.

In the remainder of this paper we focus on the sinusoidal model described above, thus leaving aside noise as well as transients. As a consequence, the following method is a lossless compression scheme for the parameters of sounds within this sinusoidal model, and not for any kind of sounds.

3. COMPRESSING THE PARAMETERS

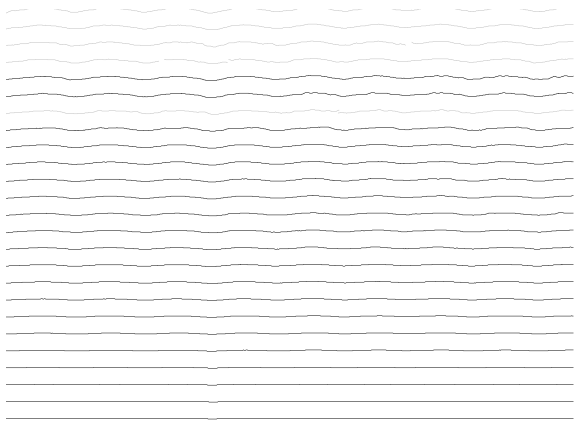
Grey [10] demonstrated that for many natural sounds the functions f_p and a_p could be simplified by piecewise-linear approximations with great data reduction by keeping only some breakpoints instead of every discrete value of the (sampled) functions. In fact, even if the consequences of this simplification are often not noticeable, they can become very audible after some transformations have been performed on the sound. Moreover, this simplification is ill-suited for sounds with vibrato or tremolo (see Figure 1). Strawn [11] also proposed to approximate the frequency and amplitude functions of the partials. Charbonneau presents in [12] a study of the perceptual effects of these data reductions on the timbre of the sounds. The compression scheme we propose in this section is a lossless compression. We consider the frequency and amplitude functions of the partials as continuous-time control signals.

3.1. Sub-Sampling

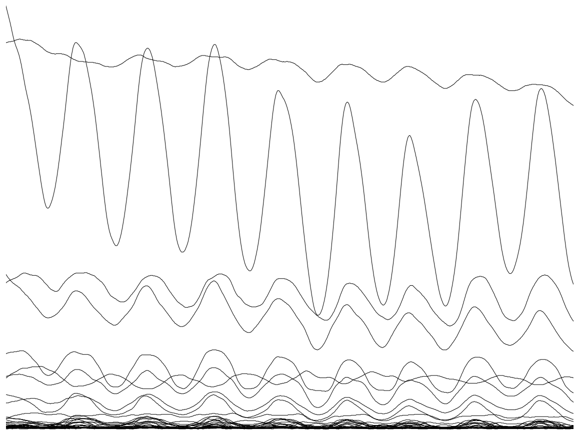
The frequencies and amplitudes of the partials are slow time-varying parameters, slow enough to avoid amplitude or frequency modulation phenomena that would modify the timbre. They can be regarded as inaudible signals controlling audible oscillations. They are indeed band-limited signals, with a maximal frequency F below the threshold of hearing, around 20 Hz. The Shannon-Nyquist theorem assures that $2F$ – only 40 here – samples per second are sufficient for the signal to be reconstructed without any error. The evolutions in time of the frequencies and amplitudes of all the partials can be safely sub-sampled. For that purpose, we use a generic resampling algorithm such as the one presented in [13].

3.2. Encoding

After the resampling of the parameters, an encoding process is performed in three steps on each parameter (frequency and amplitude) of every partial of the sound.



(a) Frequencies



(b) Amplitudes

Figure 1: *Frequencies and amplitudes of the partials of an alto saxophone as functions of time (horizontal axis), during 1 second.*

3.2.1. Delta-Encoding

First, delta encoding consists in replacing the stream of parameter samples by the one resulting from the differences between two consecutive samples, close samples leading to small differences:

$$v_1, v_2, \cdots, v_n \to v_1, (v_2 - v_1), \cdots, (v_n - v_{n-1})$$

3.2.2. Variable-Length Quantities (VLQ)

Then, variable-length quantities – also used in MIDI files [14] – allow us to store each delta-encoded value with a reduced number of bits. Each value is stored as a series of bytes which is called a variable-length quantity. Only the first 7 bits of each byte are significant. So, if we have a value requiring more than 7 bits, we have to unpack it into a series of 7-bit bytes. Of course, we will have a variable number of bytes depending upon our value. To indicate which is the last byte of the series, we leave bit 7 clear. In all of the preceding bytes, we set bit 7. So, if the value is between 0 and 127, it can be represented as one byte. More generally, if the value is between 2^{7k} and $2^{7(k+1)} - 1$, k bytes are required for its corresponding variable-length quantity.

size (in Kbytes)	sine	voice	saxophone	trumpet
temporal	86	127	246	3190
spectral	11	1077	1137	13134
resampled	<1	270	284	3284
Delta	<1	135	142	1642
VLQ	<1	30	30	660
RLE	<1	12	15	589
spectral (zip)	9	753	1051	10449

Table 1: *Some results using our compression method. The last line gives the performance of the Lempel-Ziv method (zip files).*

3.2.3. Run-Length Encoding (RLE)

Finally, a classic run-length encoding algorithm is used to compress the sequences of consecutive samples with the same value. Run-length encoding stands for the specification of elements in a list as a list of pairs giving the element and number of times it occurs in a run. This is a well-known technique widely used in mathematics and computer science. Here is an example of run-length encoding:

$$1, 1, 1, 2, 3, 3, 4, 4, 4, 4, 4 \to (1, 3), (2, 1), (3, 2), (4, 5)$$

This technique allows us to compress the list of values whenever consecutive values are the same.

3.3. Results

Concerning the compression ratio, the alto saxophone sound partly shown in Figure 1 lasts 2.85 seconds and was sampled at 44100 Hz. As a consequence, its time-domain representation (temporal model) using 16-bit quantization consists of 246 Kbytes. When the spectral analysis is done each 64 samples, this sound gives rise to an SDIF file of 1137 Kbytes (using sinusoidal tracks with 64-bit floating point values). After the resampling and delta-encoding steps, its size is only 284 Kbytes. After the VLQ step, this size drops to 30 Kbytes. At the end, after the RLE step, it is only 15 Kbytes. In this case there is a $1/75$ ratio from the SDIF information. Other examples can be found in Table 1. Moreover, we show that the well-known Lempel-Ziv method – used for example in the zip compression – performs poorly on the spectral data.

3.4. File Format

The resulting structures are not suited for implementation in the Sound Description Interchange Format (SDIF) [15, 16, 17], which is overall a succession of frames ordered in time. This is the reason why we have designed a new file format for (compressed) spectral sounds. This format is currently used in the *InSpect* analysis tool [8, 9] as the MSC format.

The detailed description of the MSC format can be found in the sources of *InSpect*. For short, this format is composed of a header followed by the compressed partials. Each partial consists

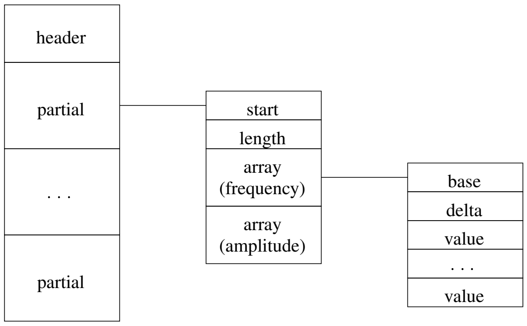


Figure 2: *The hierarchical structure of the MSC file format.*

of two values (start and length) followed by two compressed arrays, one for the frequency and the other for the amplitude of the partial. Each compressed array is composed of the stream of values of the down-sampled evolutions of the considered parameter of the partial. The first sample is called the "base", and the amplitude of the variations is called the "delta". In fact, the values (v_i) are stored relatively to the base and normalized according to the delta, that is $(v_i - \text{base})/\text{delta}$, encoded as integers using variable-length quantities (see above).

4. COMPRESSION ENHANCEMENT

It is possible to enhance this basic compression scheme.

4.1. Adaptive Sub-Sampling

First, it is possible to enhance the compression ratio by performing an adaptive sub-sampling, that is by choosing the minimal sampling rate allowed for the parameters f_p and a_p of each partial p instead of fixing an unique sampling rate for all the partials during the resampling step.

4.2. Interdependent Parameters

It is also possible to enhance the compression ratio by taking advantage of the potential interdependence of the parameters.

4.2.1. Pseudo-Harmonic Sounds

For pseudo-harmonic sounds like in Figure 1(a), we use the knowledge that the frequencies are close to multiples of the fundamental frequency. We store only the ratios to this frequency, which are close to integers.

4.2.2. Analysis of Partials Evolutions

Such relations among partials can also exist for their amplitudes, as in Figure 1(b), and can be found out by analyzing the parameters themselves and comparing the results among partials.

Performing a spectral analysis on the parameters of the partials is of great interest too. More formally, we consider the classic sinusoidal model as defined in Equations 1 and 2. Since the amplitude and frequency of each partial can be considered as band-

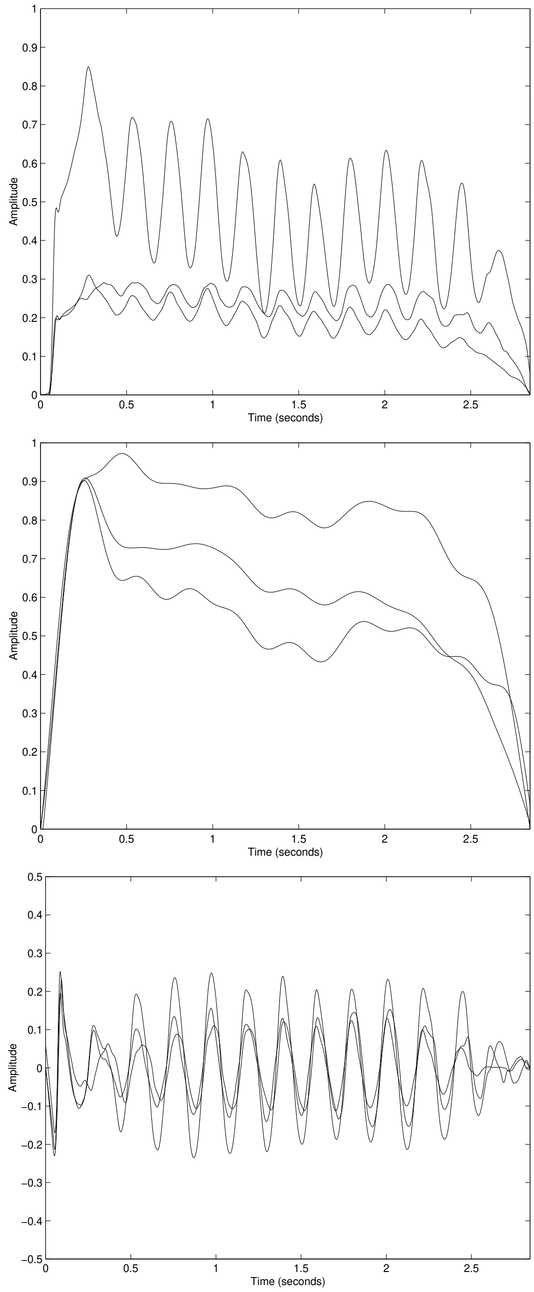


Figure 3: *The amplitudes of the first three partials of an alto saxophone (a_p, $1 \leq p \leq 3$) as functions of time (horizontal axis), decomposed here as macroscopic envelopes ($a_{a_p,0}$) and residual microscopic variations. From top to bottom are the amplitude functions a_p, the associated macroscopic envelopes $a_{a_p,0}$, and the residual microscopic variations. As a consequence, the tremolo is separated from the envelopes. This tremolo is coded in the $f_{a_p,1}$ and $a_{a_p,1}$ parameters. Since the frequency of the tremolo is the same for the three partials, the $f_{a_p,1}$ ($1 \leq p \leq 3$) values are indeed very similar.*

limited signals, we now decompose them within a similar sinusoidal model. More formally:

$$\begin{cases} a_p(t) = a_{a_p,0}(t) + \sum_{n=1}^{P_{a_p}} a_{a_p,n}(t)\, cos(\phi_{a_p,n}(t)) \\ \phi_{a_p,n}(t) = \phi_{a_p,n}(0) + 2\pi \int_0^t f_{a_p,n}(u)\, du \end{cases} \tag{3}$$

and

$$\begin{cases} f_p(t) = a_{f_p,0}(t) + \sum_{n=1}^{P_{f_p}} a_{f_p,n}(t)\, cos(\phi_{f_p,n}(t)) \\ \phi_{f_p,n}(t) = \phi_{f_p,n}(0) + 2\pi \int_0^t f_{f_p,n}(u)\, du \end{cases} \tag{4}$$

where $a_{a_p,0}$ and $a_{f_p,0}$ are extremely slow time-varying functions, band-limited to a frequency much lower than the smallest $f_{a_p,n}$ and $f_{f_p,0}$ frequencies, in practice a few Hz. These $a_{a_p,0}$ or $a_{f_p,0}$ parameters define the macroscopic variations in, respectively, amplitude or frequency – that is the "envelopes" – whereas the other parameters (for $n > 0$) reflect the microscopic variations. This decomposition is illustrated in Figure 3.

By performing a spectral analysis, we can extract from each amplitude or frequency parameter of each partial an envelope together with pseudo-partials, $(f_{a_p,n}, a_{a_p,n})$ or $(f_{f_p,n}, a_{f_p,n})$ (for $n > 0$), respectively, that are in fact the evolutions of the spectral parameters resulting from the reanalysis of the parameter of the partial. Thus the analysis of the evolutions of the partials results in other (pseudo) partials, very slow-time varying.

In this further analysis level we can find similarities among the parameters. For example, a sound with tremolo or vibrato will show such similarities among the partials. It is possible to reduce the amount of data needed to represent a sound by sharing these redundant informations.

5. CONCLUSION

We perform lossless compression with impressive ratios and the file format we have designed is very compact. It has proven to be very useful for analysis / synthesis programs based on spectral modeling. Even if we have not investigated it yet, we believe that our technique could be used for lossy but perceptually lossless compression [18], using quantization and taking advantage of psycho-acoustical phenomena such as masking [19] to reduce the number of partials to be stored.

6. ACKNOWLEDGMENTS

This research was carried out in the context of the SCRIME (*Studio de Création et de Recherche en Informatique et Musique Electroacoustique*) and was supported by the *Conseil Régional d'Aquitaine*, the *Ministère de la Culture*, the *Direction Régionale des Actions Culturelles d'Aquitaine*, and the *Conseil Général de la Gironde*.

7. REFERENCES

[1] Marie-Hélène Serra, *Musical Signal Processing*, chapter Introducing the Phase Vocoder, pp. 31–90, Swets & Zeitlinger, Lisse, the Netherlands, 1997.

[2] Robert J. McAulay and Thomas F. Quatieri, "Speech Analysis / Synthesis Based on a Sinusoidal Representation," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 34, no. 4, pp. 744–754, 1986.

[3] Kelly Fitz and Lippold Haken, "Sinusoidal Modeling and Manipulation Using Lemur," *Computer Music Journal*, vol. 20, no. 4, pp. 44–59, 1996.

[4] Xavier Serra and Julius O. Smith, "Spectral Modeling Synthesis: A Sound Analysis / Synthesis System Based on a Deterministic plus Stochastic Decomposition," *Computer Music Journal*, vol. 14, no. 4, pp. 12–24, 1990.

[5] Xavier Serra, *Musical Signal Processing*, chapter Musical Sound Modeling with Sinusoids plus Noise, pp. 91–122, Swets & Zeitlinger, Lisse, the Netherlands, 1997.

[6] Tony S. Verma and Teresa H.Y. Meng, "An Analysis / Synthesis Tool for Transient Signals that Allows a Flexible Sines+Transients+Noise Model for Audio," in *Proceedings IEEE ICASSP*, May 1998.

[7] Tony S. Verma and Teresa H.Y. Meng, "Time Scale Modification Using a Sines+Transients+Noise Signal Model," in *Proceedings of the Digital Audio Effects Workshop (DAFx'98)*, Barcelona, 1998, pp. 49–52.

[8] Sylvain Marchand and Robert Strandh, "Inspect and ReSpect: Spectral Modeling, Analysis and Real-Time Synthesis Software Tools for Researchers and Composers," in *Proceedings ICMC*, Beijing, October 1999, pp. 341–344.

[9] Sylvain Marchand, "InSpect Software Package," Online. URL: `http://www.scrime.u-bordeaux.fr`, 2000.

[10] John M. Grey, *An Exploration of Musical Timbre*, Ph.D. thesis, Department of Music, Stanford University, 1975.

[11] John Strawn, "Approximation and Syntactic Analysis of Amplitude and Frequency Functions for Digital Sound Synthesis," *Computer Music Journal*, vol. 4, no. 3, 1980.

[12] G. R. Charbonneau, "Timbre and Perceptual Effects of Three Types of Data Reduction," *Computer Music Journal*, vol. 5, no. 2, pp. 10–19, 1980.

[13] Julius O. Smith and Phil Gossett, "A Flexible Sampling-Rate Conversion Method," in *Proceedings IEEE ICASSP*, San Diego, March 1984, vol. 2, pp. 19.4.1–19.4.2.

[14] MIDI Manufacturers Association, "The Complete MIDI 1.0 Detailed Specification - Standard MIDI Files 1.1," Tech. Rep., MMA, URL: `http://www.midi.org`, 1996.

[15] Matthew Wright, Amar Chaudhary, Adrian Freed, Sami Khoury, and David Wessel, "Audio Applications of the Sound Description Interchange Format Standard," *107th AES Convention*, September 1999.

[16] CNMAT, "SDIF: Sound Description Interchange Format," Online. URL: `http://cnmat.CNMAT.Berkeley.EDU/SDIF/`, 2000.

[17] IRCAM, "SDIF – Sound Description Interchange Format," Online. URL: `http://www.ircam.fr/equipes/analyse-synthese/sdif/`, 2000.

[18] Nikil Jayant, James Johnston, and Robert Safranek, "Signal Compression Based on Models of Human Perception," in *Proceedings IEEE*, 1993, vol. 81, pp. 1385–1421.

[19] Guillermo Garcia and Juan Pampin, "Data Compression of Sinusoidal Modeling Parameters Based on Psychoacoustic Masking," in *Proceedings ICMC*, Beijing, October 1999, pp. 40–43.