High Precision Fourier Analysis of Sounds using Signal Derivatives

Myriam Desainte-Catherine (myriam@LaBRI.U-Bordeaux.Fr)
Sylvain Marchand (sm@LaBRI.U-Bordeaux.Fr)

May 1, 1998

Abstract

This article introduces an original method that greatly improves Fourier analysis precision both in frequency and amplitude, but also in time, thus minimizing the problem of the trade-off of time versus frequency with the classical short-time Fourier transform. This method is of great interest for extracting spectral modeling parameters from existing sounds and, after a detailed theoretical presentation, practical results obtained from implementation are exposed at the end of this paper.

1 Introduction

Digital sound synthesis and manipulation require a good model of sound. Spectral models - parametrizing sound at the basilar membrane in the ear - give general representation well-suited for meaningful sound transformations. In order to faithfully imitate or transform existing sounds, an analysis method is necessary to extract the model parameters from them. The accuracy of the analysis method is extremely important since the perceived quality of the resulting sounds depends mainly on it.

This article introduces an original technique that greatly improves Fourier analysis precision both in frequency and amplitude, but also in time, thus minimizing the effects of the well-known trade-off of time versus frequency. More precisely, this method extends the short-time Fourier transform by also considering signal derivatives, and effectively leads to efficient spectral parameters extraction, thus allowing precise modifications of the inner structures of sound.

After introducing in section 2 the spectral sound model which is considered in the rest of this article, section 3 points out limitations of the main existing analysis method for extracting the model parameters: the short-time Fourier transform. Then, an enhanced analysis method is exposed in section 4, and section 5 presents a discrete version of practical interest. It provides accurate frequency information with great time resolution, which is used then to obtain accurate amplitude information too. Finally, some practical results of the application of this method on synthetic and natural sounds are presented in section 6.

2 Sound Model

In order to synthesize new digital sounds or manipulate existing ones using a computer, a formal representation is needed for audio signals. A sound model denotes such a mathematical representation. It should be as general as possible so that most of the sounds can be faithfully reproduced and transformed in a natural and musically expressive way. This section presents the sound model considered in this article.

2.1 Spectral versus Temporal

The simplest way to represent an audio signal is to consider its amplitude as a function of time $a(t)$, where t is time expressed in seconds. This sound representation as a time-signal is often called the temporal model. But it is well known that the simplicity of this model has severe drawbacks as soon as the inner structures of sound have to be manipulated. In fact the only musical parameters that can easily be modified are the volume (by scaling amplitude axis) and the pitch (by scaling time axis), and this last transformation, which is described in [SG84] and still used by many samplers, may also modify sound timbre in an unnatural way. Of course deeper transformations can also be performed in the time domain, but their mathematical foundations are not so intuitive and most of them are silently switching to spectral domain by the mean of a Fourier transform...

The sound model which is considered in this article is a spectral model, dealing with both frequencies and amplitudes as functions of time. It provides a general representation for sound, allowing intuitive and expressive transformations according to the perception.

2.2 Spectral Model

Xavier Serra and Julius O. Smith III have proposed in [SS90] a spectral model based on a deterministic plus stochastic decomposition. This is mostly the model which is considered here, expect that the equations have been reformulated for homogeneity purpose and some hypotheses have been modified. These modifications will be introduced when needed. This sound model decomposes any audio signal a as the sum of two signals:

$$a(t) = d(t) + s(t)$$

where

- $d(t)$ is the deterministic part (narrowband component),
- $s(t)$ is the stochastic part (broadband component).

The deterministic part is the sum of sinusoidal oscillators whose frequency and amplitude evolve in a slow time-varying manner. Such a low time-varying oscillator is commonly called a *partial*. More formally:

$$d(t) = \sum_{p=1}^{P} osc(f_p(t), a_p(t))$$

where P is the number of partials,

$$osc(f_p(t), a_p(t)) = a_p(t) \cdot cos(\phi_p(t))$$

with

$$\frac{d\phi_p}{dt}(t) = 2\pi f_p(t) \quad \text{i.e.} \quad \phi_p(t) = \phi_p(0) + 2\pi \int_0^t f_p(u) \cdot du$$

The functions f_p, a_p, and ϕ_p are respectively the frequency, amplitude, and phase of the p-ieth partial. The phase at the origin $\phi_p(0)$ will be ignored during analysis and arbitrary set to 0 for resynthesis[1].

The hypothesis that partials frequencies and amplitudes evolve slowly is indeed very important. For example, given any sound a, the solution $P = 1$, $f_1(t) = 0$, $a_1(t) = a(t)$ trivially verifies the model equations, but since a_1 is not slow time-varying such a solution is ruled out by this hypothesis. However, the definition of "slow time-varying" has to be precised. This will be done in section 4.

Although it is often useful to distinguish two types of partials - harmonic and nonharmonic partials - depending on whether or not they have equally spaced frequencies [MP80], this distinction will not be done here. It can really be of great interest for pitch detection, and pitch-synchronous analysis methods can take advantage of this. However, this is not the case of the technique proposed in this article.

Stochastic part is what remains after deterministic part has been substracted from the original signal and is generally called noise. It involves random and statistical properties which are beyond this article, but stochastic part can be fully described by its power spectral density which gives the expected signal power versus frequency. However, noise analysis and resynthesis are out of the scope of this paper, which focuses on an analysis method for extracting parameters for the deterministic part.

2.3 Restricted Spectral Model

The rest of this paper will focus on sounds where stochastic part can be neglected, so that $s(t) = 0$ can be assumed. Thus, let's take in the rest of this paper:

$$a(t) = \sum_{p=1}^{P} osc(f_p(t), a_p(t)) \tag{1}$$

In pratice, this restriction means that considered sounds should have low noise, which is true for many clear natural sounds, eventually after their attack phase.

Another restriction is that partials have to be spaced enough in frequency: given any sound a, it must exist a minimal distance $d > 0$ so that:

$$min_{i \neq j, t}\{|f_j(t) - f_i(t)|\} > d$$

[1] This choice can be done according to psychoacoustic experiments which are beyond the present article.

For example, this hypothesis prevents two partials frequencies from “crossing”. This is a reasonable hypothesis verified for almost every monophonic natural sound. The reasons why it is needed will be discussed in section 5.

Grey [Gre75] demonstrated that for many natural sounds the functions f_p and a_p could be simplified by piecewise-linear approximation with great data reduction by keeping only some breakpoints instead of every discrete value of the functions. Since data reduction is not important here, this approximation will not be done. In fact, even if its consequences are often not noticeable, they can become very audible after some transformation has been performed on the sound.

This restricted spectral model is often called a sinusoidal model. Many sinusoidal models have been proposed in recent years, and have demonstrated their practical interest in software implementations like Lemur [FH96] and SMS [Ser97b], thanks to the always increasing power of nowadays computers.
However, sinusoidal models are derived from the work of Jean Baptiste Fourier during the ninetieth century. Next section presents one of the most famous method used to extract model parameters from existing sounds.

3 Short-Time Fourier Analysis and Partial Tracking

Additive synthesis [Moo77] is the original spectrum modeling technique. It is rooted in Fourier's theorem, which states that any periodic function can be modeled as a sum of sinusoids at various amplitudes and harmonic frequencies. The spectral model considered here is indeed very similar to additive synthesis models, aside from the way partials may vary in time. Another important point is that their phases are recomputed from scratch and not given by analysis.

The most famous analysis method for additive synthesis is probably the phase vocoder. Very good introductory texts on the phase vocoder can be found for example in [Moo78], [Por80], [Dol86], or [Ser97a]. It is mainly an implementation of the short-time Fourier transform [All77]. This section makes a short summary of the principles of this transformation and points out its main limitations.

3.1 Principles

The Fourier transform converts the temporal signal (amplitude versus time) into a spectral representation (amplitude versus frequency). It gives the frequency image of the whole sound, and the entire signal is averaged into a single spectrum. This spectrum coincides with our perception only for stationnary sounds. Because most of sounds evolve in time, sound analysis algorithms must produce time-varying results. Those algorithms generally repeat the same procedures on successive short sections of sound. This kind of processing is called short-time analysis. Among the most famous short-time analysis is the short-time Fourier transform, which gives a time-dependent version of the Fourier transform. It produces a serie of short-time spectra taken on successive temporal frames, which are small pieces of temporal signal. In practice this is a discrete signal resulting from uniform sampling with sampling rate $\mathcal{R}$. Then for each frame x

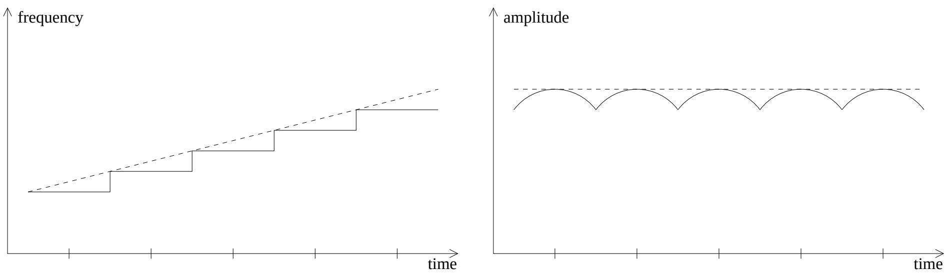

Figure 1: Original (dashed) versus Fourier analyzed (plain) frequency and amplitude evolutions for a single sinusoidal oscillator whose frequency is linearly increasing while its amplitude remains constant. (The marks on the time axis indicate when the oscillator frequency changes of bin.)

of N consecutive samples, its (discrete) Fourier transform $\mathcal{X}$ is computed:

$$\mathcal{X}[m] = \frac{1}{N} \sum_{n=0}^{N-1} x[n] \cdot e^{-j\frac{2\pi}{N}nm} \tag{2}$$

In fact a (discrete) windowing function w is always applied (multiplied) to x. If $w[k]$ is 1 for every $0 \leq k < N$ (and 0 outside) - like in equation 2 - w is called the rectangular window (because the shape of its graph looks like a rectangle). This is probably the worst analysis window... The rest of this section will focus on the (short-time) power spectra obtained from the short-time Fourier transform.

3.2 Limitations

The analysis window has a great impact on analysis precision both in frequency and amplitude. Its effects on the spectrum must be reduced as much as possible. The shape and width of the analysis window are key factors. An exhaustive discussion about analysis windows is beyond this paper. Information about analysis windows can be found in [Har78] and [OS89]. However section 5 will show that the Hann window is convenient.

Figure 1 provides a clear synthetic view of two phenomena. It shows the result of the short-time Fourier transform on a single sinusoidal oscillator whose frequency is linearly increasing while its amplitude remains constant. The analyzed frequency curve is not a line as it should be, but a sort of stairs, due to spectrum sampling (frequency imprecision). And the analyzed amplitude curve is not flat, i.e. not a constant, but a succession of bumps, due to window mainlobe shape (amplitude imprecision).

3.2.1 Frequency Imprecision

The power spectrum is the magnitude of the spectrum. Let's note S the power spectrum of a signal s, and $S(f)$ its value at frequency f. Since in practice the discrete Fourier transform is used, this spectrum is sampled from 0 Hz (DC

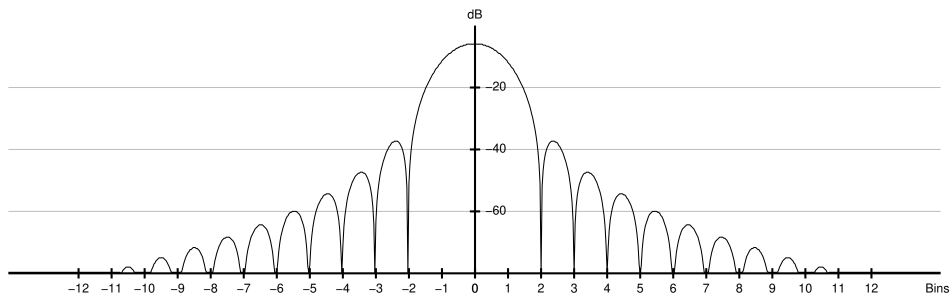

Figure 2: Hann window power spectrum (W_{Hann}).

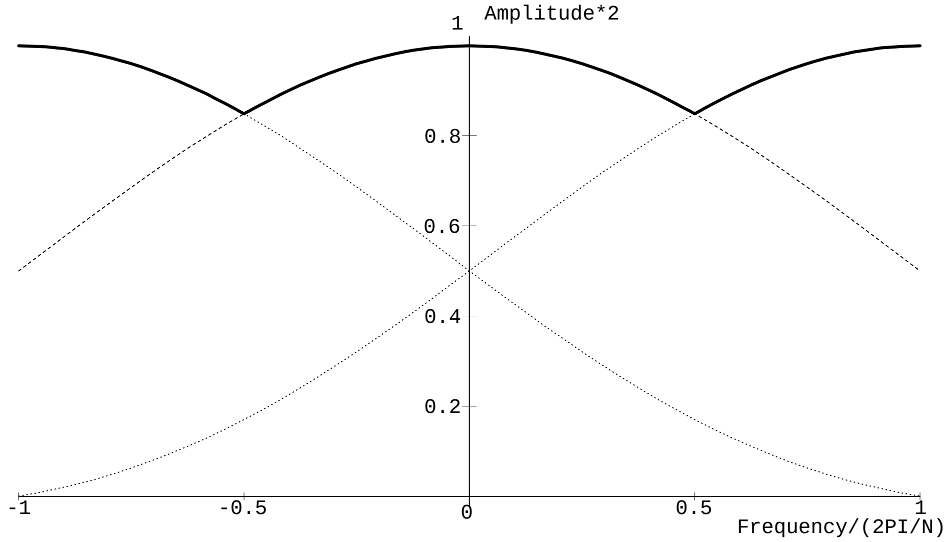

Figure 3: Hann window lobes.

component) to $\frac{\mathcal{R}}{2}$ Hz (Nyquist frequency) by steps of $\frac{\mathcal{R}}{N}$ Hz. Let's note $S[k]$ its value at frequency $k\mathcal{B}$ where $\mathcal{B} = \frac{\mathcal{R}}{N}$. The Fourier transform can be regarded as a fixed filter bank, and the instantaneous amplitude and frequency are computed for each of the N channel filters or bins.

The first limitation comes from the fact that frequency precision is inversely proportional to N. A good frequency precision requires a small $\mathcal{B}$, i.e. a large N, which leads to poor precision in time (since N is the analysis window width in samples). On the contrary, a good time resolution leads to a poor frequency precision. This is the well-known trade-off of time versus frequency of the classical Fourier analysis.

3.2.2 Amplitude Imprecision

The spectrum of the $w \cdot x$ product is the convolution $W * X$. If x is a pure sinusoidal oscillation with amplitude 1 and frequency f, X is a spectral ray: $X(f) = 1$ and $X(f') = 0$ for $f' \neq f$. But since W is like in Figure 2, $W * X$ is far from being a spectral ray as it should be. More precisely, to obtain a spectral

rectangular	Bartlett (triangular)	Hann	Hamming	Blackman
36%	19%	15%	18%	12%

Figure 4: Amplitude imprecision for some windows.

ray W should be as close to an impulse ($W(0) = 1$ and $W(f) = 0$ for $f > 0$) as possible, which is impossible with a finite-duration analysis window. So in practice W often consists of "lobes", and it is only required that the sidelobes amplitudes are negligible compared to the mainlobe amplitude. Unfortunately, this sidelobes/mainlobe ratio is inversely proportional to mainlobe width, which determines the frequency resolution of the window.

A good frequency resolution requires a small mainlobe width, but because of the shape of the lobe, the magnitude of the spectrum is distorted as shown in Figure 1. This is the reason why the amplitude a sinusoidal oscillator is so distorted as its frequency goes from bin to bin (see Figure 3). Any good analysis method should take care of this phenomenon. Of course such little deformations can't generally be heard, but they may become dramatically audible as soon as some transformations are performed on the sound. Figure 4 shows that the amplitude imprecisions for the most common analysis windows are quite important. In order to minimize this imprecision, a "flat" mainlobe is necessary. But this is possible only with a very large mainlobe. So there is a sort of trade-off of amplitude precision versus frequency precision.

Accurate short-time spectral analysis is extremely important since it is the first step for a key technique called partial tracking. It consists in following the evolutions of power spectrum maxima in time. This technique is used in famous software packages like AudioSculpt [IRC96], Lemur [FH96] [MQ86], PARSHL [SS87], and SMS [SS90]. So many of them suffer from the short-time Fourier analysis limitations shown above. Next section introduces an original technique that greatly reduce these limitations.

4 FT^n: Fourier Transform using n Signal Derivatives

Position, speed, but also acceleration are key parameters in cinematic. There is a deep analogy between these parameters and respectively phase ϕ_p, frequency $f_p = \frac{d\phi_p}{dt}$, and also frequency derivative $\frac{df_p}{dt} = \frac{d^2\phi_p}{dt^2}$. In this article both frequency and amplitude are slow time-varying parameters, so during a single analysis window of the short-time Fourier transform $\frac{df_p}{dt}$ and $\frac{da_p}{dt}$ are close to 0. This section shows that under such conditions using the first n signal derivatives can improve Fourier analysis precision both in frequency and amplitude. The idea behind this technique is extremely simple: derivating a sine gives a sine, with different phase but same frequency. More formally, signal derivatives must be examined.

4.1 Continuous Signal Derivatives

Let's consider a single oscillator:

$$o_p(t) = osc(f_p(t), a_p(t)) = a_p(t) \cdot cos(\phi_p(t))$$

and then let's calculate its first derivative:

$$\frac{do_p}{dt}(t) = \frac{d}{dt}(a_p(t) \cdot cos(\phi_p(t))) = a_p(t) \cdot \frac{d}{dt}(cos(\phi_p(t))) + \frac{da_p}{dt}(t) \cdot cos(\phi_p(t))$$

Since a_p is slow time-varying $\frac{da_p}{dt} = 0$ can be assumed, so:

$$\frac{do_p}{dt}(t) = a_p(t) \cdot \frac{d}{dt}(cos(\phi_p(t))) = -a_p(t) \cdot \frac{d\phi_p}{dt}(t) \cdot sin(\phi_p(t))$$

But $\frac{d\phi_p}{dt}(t) = 2\pi f_p(t)$ (see section 2), so:

$$\frac{do_p}{dt}(t) = -a_p(t) \cdot (2\pi f_p(t)) \cdot sin(\phi_p(t)) = 2\pi \cdot a_p(t) \cdot f_p(t) \cdot cos(\phi_p(t) - \frac{\pi}{2}) \quad (3)$$

More generally, it is easy to verify by induction that:

$$\frac{d^k o_p}{dt^k}(t) = a_p(t) \cdot (2\pi f_p(t))^k \cdot cos(\phi_p(t) + (-k \cdot \frac{\pi}{2}))$$

Finally, since derivation is a linear operation and from equation 1, the k-ieth signal derivative is:

$$\frac{d^k a}{dt^k}(t) = \sum_{p=1}^{P} a_p(t) \cdot (2\pi f_p(t))^k \cdot cos(\phi_p(t) + (-k \cdot \frac{\pi}{2})) \quad (4)$$

4.2 Analysis Precision from Derivation

Let's note FT^k the Fourier Transform of the k-ieth signal derivative $\frac{d^k a}{dt^k}$ $(k \geq 0)$, and $FT^k(f)$ its amplitude (magnitude) at frequency f. The intuition behind equation 4 is that there should be a maximum in every FT^i $(i \geq 0)$ power spectrum for each partial p. Then the underlying idea is to use FT^i power spectra for different values of i in order to determine the exact frequencies of the set of partials. When only the first n derivatives of a are used, let's call this approach n-order derivative Fourier analysis[2].

4.2.1 Determining the Frequency

A consequence of equation 4 is that for each partial p there is a maximum in every FT^i at frequency f_p (assuming $a_p > 0$ and $f_p > 0$) and:

$$f_p = \frac{1}{2\pi} \cdot \frac{FT^{i+1}(f_p)}{FT^i(f_p)} \quad (5)$$

[2] Since $\frac{d^0 a}{dt} = a$, the standard Fourier transform coincides with the zero-order derivative Fourier transform FT^0.

Although this equation may seem of poor interest since it requires the knowledge of f_p, its discrete version presented in section 5 turns out to be of great practical interest. The reason is that the sampling of the FT^i spectra in practice leads to an approximated value f_p^0 of the frequency f_p. The exact value can be then recovered from its approximation using the discrete version of equation 5.

It is extremely important to note that the effects of any analysis window are the same on both $FT^i(f_p)$ and $FT^{i+1}(f_p)$ (as soon as the same analysis window is used to compute these two spectra). These effects are compensating thanks to the division in the preceding equation.

4.2.2 Correcting the Amplitude

Of course in order to get precise maxima in FT^0 spectrum (sharp spectral peaks), an analysis window is needed. Section 3 has pointed out the consequences of windowing on magnitude spectrum. Fortunately, the exact amplitude a_p of partial p can easily be recovered from the approximative analyzed value $a_p^0 = FT^0(f_p)$ since:

$$a_p = \frac{a_p^0}{W(0)} \tag{6}$$

Adjusting phase information is also possible as soon as precise frequency values are available, since for a small time variation Δ_t, the model equations assure that $\phi_p(t + \Delta_t) \simeq \phi_p(t) + 2\pi f_p(t)\Delta_t$. Anyway, this consideration is left aside in this article since recovering accurate phase information from analysis is not important for the considered model.

Next section describes the discrete version of the FT^n technique for $n = 1$ and shows its practical interest for extracting accurate frequency and amplitude. The choice of analysis window is also discussed.

5 DFT^1: Discrete FT^1

The FT^n transform requires the first n signal derivatives. This section shows that even without any analog device providing such information, the discrete version of FT^1 can be realized. But since signal derivative is not usually recorded together with signal, it has to be calculated from digital signal after the sampling phase. The discrete signal is uniformly sampled with sampling rate $\mathcal{R}$. For all figures and practical examples in the rest of this article the CD-audio sampling rate $\mathcal{R} = 44100Hz$ will be used.

5.1 Discrete Signal Derivatives

The first derivative $\frac{da}{dt}$, also noted a', is mathematically defined by:

$$a'(t) = \lim_{\epsilon \to 0} \frac{a(t+\epsilon) - a(t)}{\epsilon}$$

In fact, left and right derivatives for ϵ respectively negative or positive are:

$$a'_-(t) = \lim_{\epsilon \to 0_-} \frac{a(t+\epsilon) - a(t)}{\epsilon} = \lim_{\eta \to 0_+} \frac{a(t) - a(t-\eta)}{\eta} \quad (\eta = -\epsilon)$$

$$a'_{+}(t) = \lim_{\epsilon \to 0_{+}} \frac{a(t+\epsilon) - a(t)}{\epsilon}$$

Since signal a is a $\mathcal{C}^{\infty}$ function, left and right derivatives are equal, i.e.:

$$a'_{-}(t) = a'_{+}(t) = a'(t)$$

In practice, the audio signal a is discrete, $a[i]$ representing $a(i \cdot \frac{1}{\mathcal{R}})$, and the smallest (non-zero) possible $|\epsilon|$ for the derivative computation is the sampling period $\frac{1}{\mathcal{R}}$. So let's consider the following approximations:

$$a'_{-}[i] = (a[i] - a[i-1]) \cdot \mathcal{R} \quad \text{and} \quad a'_{+}[i] = (a[i+1] - a[i]) \cdot \mathcal{R}$$

This time $a'_{+}[i] \neq a'_{-}[i]$, but:

$$a'_{+}[i] = (a[i+1] - a[i]) \cdot \mathcal{R} = a'_{-}[i+1]$$

$$a'_{-}[i] = (a[i] - a[i-1]) \cdot \mathcal{R} = a'_{+}[i-1]$$

so appart a one-sample time translation[3], they are the same discrete functions. Let's arbitrary take left derivative approximation as an approximation for the derivative itself (taking $a[i] = 0$ for $i < 0$):

$$a'[i] = \mathcal{R} \cdot (a[i] - a[i-1]) \tag{7}$$

Equation 7 has a form which is quite classical in digital filters theory [Moo90]. More precisely, it defines a (linear phase) high-pass filter. Its equation is:

$$y[n] = \mathcal{R} \cdot x[n] - \mathcal{R} \cdot x[n-1]$$

its transfer function can be easily calculated:

$$H(z) = \mathcal{R} \cdot (1 - z^{-1})$$

and finally its gain is:

$$|H(e^{j\omega})| = \mathcal{R} \cdot \sqrt{2(1 - cos(\omega))} \quad \text{with} \quad \omega = \frac{2\pi f}{\mathcal{R}}$$

The derivation is a linear operation that can also be considered as a filtering operation with a $2\pi f$ gain (see equation 3), that is $\mathcal{R}\omega$, which is a theoretical gain quite different from the pratical one $|H(e^{j\omega})|$, specially for great values of ω. Figure 5 illustrates this phenomenon. Fortunately this difference can be corrected by multiplying the power spectrum of the signal "derivative" approximation obtained from equation 7 by the F scaling factor defined in equation 8.

$$F(\omega) = \frac{\omega}{\sqrt{2(1 - cos(\omega))}} \quad \text{with} \quad \omega = \frac{2\pi f}{\mathcal{R}} \tag{8}$$

It is easy to show that sharp spectral peaks are roughly conserved by slowly growing gain. Figure 6 illustrates this phenomenon.

[3]This one-sample translation has a negligible impact on the power spectrum of the signal derivative, which is the only thing considered by the method.

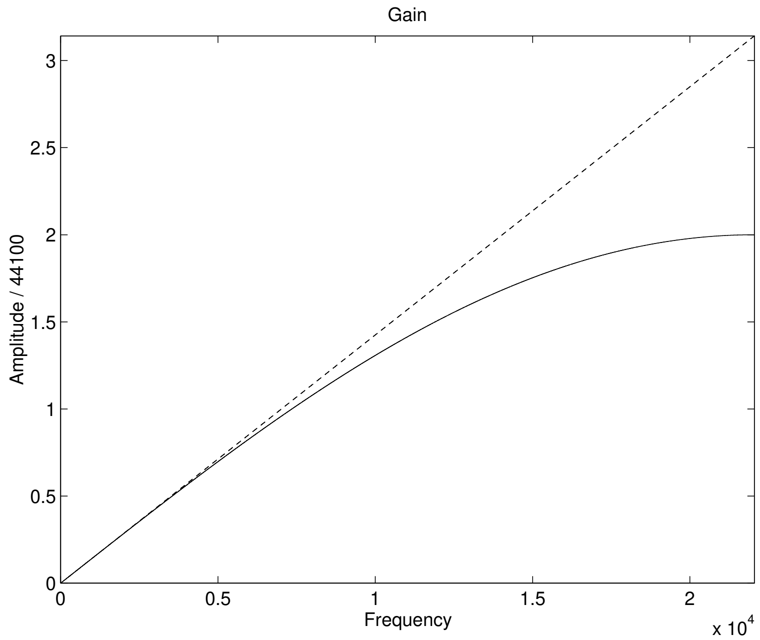

Figure 5: Practical gain $|H|$ (plain) versus theoretical derivation gain (dashed).

5.2 Analysis Precision from Discrete Derivation

5.2.1 Accurate Frequency

A discrete equivalent of equation 5 for $i = 0$ is:

$$f_p = \frac{1}{2\pi} \cdot \frac{DFT^1[m_p]}{DFT^0[m_p]} \tag{9}$$

where m_p is the index of the maximum in DFT^0 corresponding to frequency f_p. More precisely, $m_p = \lfloor f_p \cdot \frac{N}{\mathcal{R}} \rfloor$ and $m_p\mathcal{B} \leq f_p < (m_p + 1)\mathcal{B}$ (if this is not the case for the calculated f_p, then the DFT^1 analysis has failed for this frequency. In practice, 90 to 100 percent of success are common).

The exact frequency of every partial can be determined by considering equation 9 for every maximum m in DFT^0. Since only the first derivative of a is used, let's call this method the first order derivative Fourier analysis.

Again, as in the continuous case, it is extremely important to note that the effects of any analysis window are the same on both $DFT^0[m_p]$ and $DFT^1[m_p]$ (as soon as the same analysis window is used to compute these two spectra). These effects are still compensating thanks to the division in the preceding equation.

5.2.2 Accurate Amplitude

The exact amplitude of each partial can then be determined from the approximate amplitude a_p^0 and frequency f_p^0 together with the accurate frequency f_p

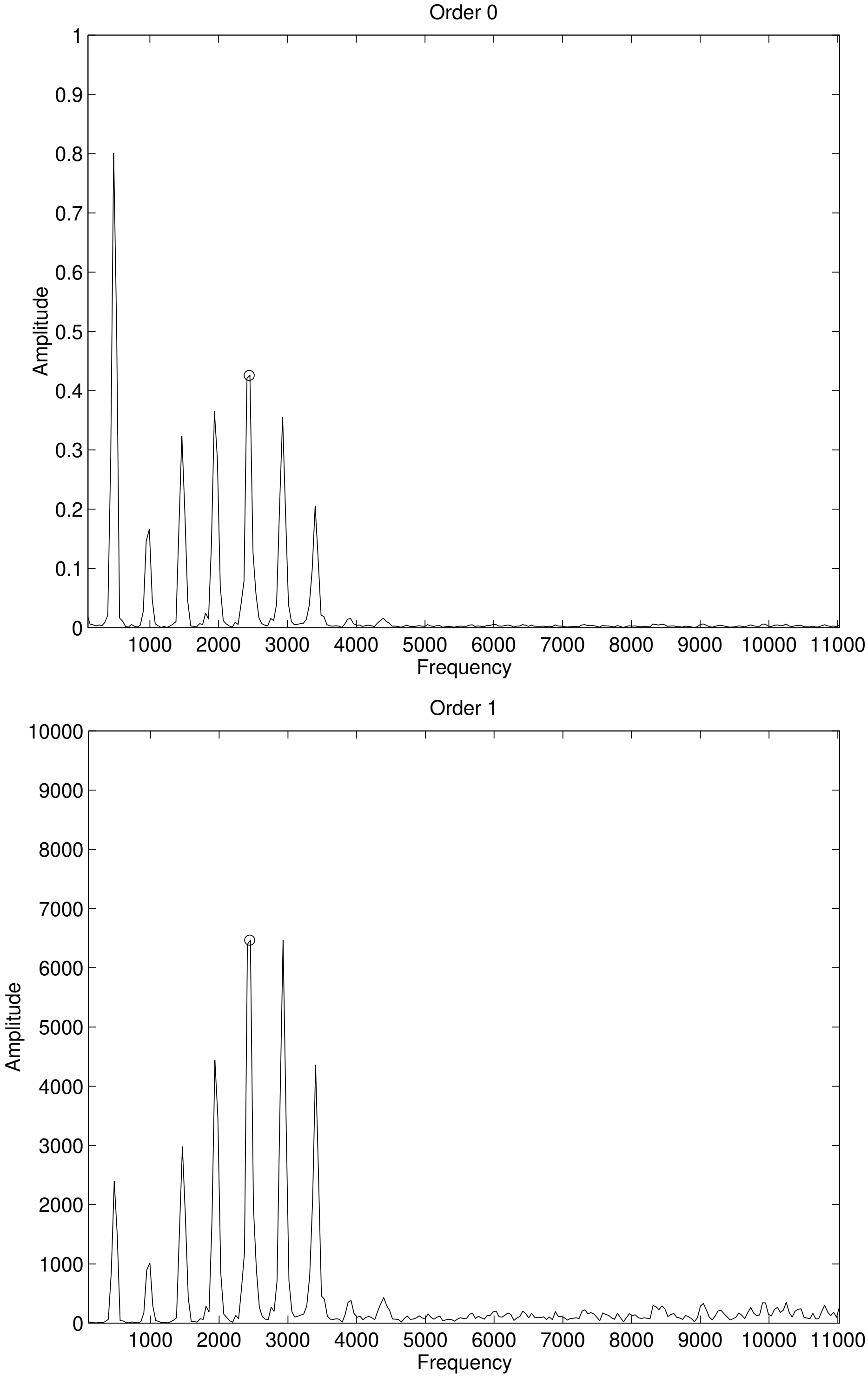

Figure 6: DFT^0 and DFT^1 power spectra.

type	one-peak	side/mainlobe ratio	mainlobe width
rectangular	yes	-13 dB	$2\mathcal{B}$
Bartlett (triangular)	no	-26 dB	$4\mathcal{B}$
Hann	yes	-31 dB	$4\mathcal{B}$
Hamming	no	-42 dB	$4\mathcal{B}$
Blackman	no	-58 dB	$6\mathcal{B}$

Figure 7: Some analysis windows characteristics.

computed before using the discrete version of equation 6:

$$a_p = \frac{a_p^0}{W(\frac{|f_p - f_p^0|}{\mathcal{B}})} \tag{10}$$

It also requires the knowledge of the (continuous) power spectrum W of the analysis window.

5.3 Choosing the Analysis Window

The choice of the analysis window is very important. Although an exhaustive discussion about analysis windows is beyond this paper, Figure 7 provides a short summary of the characteristics of the most common analysis windows. Basic knowledge on analysis windows is required to fully understand the discussion below. Information about analysis windows can be found for example in [Har78] and [OS89].

First, the type of the analysis window must be fixed, according to three principal requirements. First of all, a pure sinusoidal signal must always lead to a single local extrema in the magnitude spectrum, in order to have only one peak during the partial tracking phase. More formally, for all $0 \leq f < \mathcal{B}$, the $(W(f + k\mathcal{B}))_{k \geq 0}$ sequence must be decreasing. Among all the windows that are considered in Figure 7, only the rectangular and Hann windows have this "one-peak" property. The second requirement is that the sidelobes can be neglected in comparison to the mainlobe. This is the case for the hann window, but not for the rectangular one. Then the third requirement is that the mainlobe must be narrow, in order to have good frequency resolution. Again the Hann window turns out to be a good candidate.

Then the size of the analysis window must be precised. On the one hand N has to be large enough so that $\mathcal{B} < min_{i \neq j}(|f_j - f_i|)$, which is always possible because of the model restrictions assumed in section 2. The reason behind this condition is that two frequencies must lay in different Fourier transform bins, because $\frac{f_i a_i + f_j a_j}{a_i + a_j} \neq f_i + f_j$ and this would cause problems with equation 9. On the other hand, the analysis window should remain small enough, so that the frequencies and amplitudes may not vary too much during window duration. Practical values are precised in section 6.

5.4 Analysis Algorithm

An important problem arises with order 2 analysis, due to computation imprecisions during the calculation of the second signal derivative. In fact, for order 1 with 44100 Hz sampling rate, 16-bit samples are just precise enough, since the imprecision resulting from 16-bit quantization ($\frac{1}{65536}$) is multiplied by $\mathcal{R} = 44100Hz$, and becomes then very significant. A 24-bit quantization should be a solution. Experiments with high-quality sampling are in progress. For now, only the DFT^1 method has been successfully implemented.

Here is an unformal description of a standard analysis algorithm using the DFT^1 method. It is given as a starting point for a practical implementation. For each frame of the temporal signal the following operations have to be sequentially done:

1. apply windowing function to a
2. perform DFT^0 (discrete Fourier transform of this current frame of a)
3. compute a' using equation 7
4. apply (same) windowing function to a'
5. perform DFT^1 (discrete Fourier transform of this current frame of a')
6. scale DFT^1 power spectrum by correcting factor F (see equation 8)
7. for each m index of a local maximum in DFT^0 do:
   (a) compute accurate frequency using equation 9
   (b) compute accurate amplitude using equation 10
   (c) add couple $(frequency, amplitude)$ to the result list for the current frame

Since a' can be incrementaly computed at each frame and since DFT^1 values are needed only for the local maxima of DFT^0, two immediate optimizations can be performed, thus saving both space and time. Moreover if analysis window size is a power of 2, using the Fast Fourier Transform (FFT) to implement the Discrete Fourier Transform (DFT) makes this algorithm even faster.

Of course the partials must then be tracked from frame to frame in order to recover which $(frequency, amplitude)$ couple belongs to which partial p.

Next section presents some results obtained with a direct implementation of the DFT^1 method with the previous algorithm, and a very simple peak-tracking stategy.

6 Examples

The optimized version of the algorithm given at the end of section 5 has been implemented in a software package called *InSpect*. This section presents some of its results on synthetic and natural sounds. The Hann analysis window has been used, and the DFT^1 method has proven to be very accurate in practice too.

6.1 Base Case: Sinusoidal Oscillator

Even with very small analysis windows width - like 256 points (less than 6 ms analysis time) - the evolution of the partial shown in Figure 1 is almost perfectly recovered. Such a result would have been impossible to achieve with standard Fourier analysis since a large analysis window is needed to have a great frequency precision, and then the time resolution is so bad that the evolution of the frequency with time can not be successfully recovered. The effect of windowing on the amplitude has also almost totally been cancelled.

The example in Figure 8 shows how the analysis window width should be chosen: it must be as small as possible. The only condition is that two frequencies must lay in two different Fourier transform bins. If the analysis window is too small, it won't be the case. And if it is too large, the spectrum is averaged by the Fourier transform and both methods perform poorly.

Of course, the example of a single oscillator is synthetic. But this is in fact a reference example for analysis. Most of natural sounds are indeed made of a sum of partials, and since the analysis process is a linear operation, it should also precisely recover the time evolutions of their parameters.

6.2 Results on Natural Sounds

The DFT^1 method has also been successfully used to analyze natural sounds with low noise level. An exhaustive presentation of these results is out of the scope of the current paper. However, the method succeeds even with polyphonic sounds (see [Cas82]) and voices with deep vibrato. It is well-known that sounds with vibrato are hard to analyze with the standard short-time Fourier transform. The reason of this difficulty is illustrated in Figure 8.

The examples which are represented in Figure 9 are the results of the DFT^1 method in comparison to the standard short-time Fourier analysis on a voice of a soprano singer. The best result with standard Fourier analysis was obtained for an analysis window of 4096 points (93 ms analysis time). This result is at the top of Figure 9. With the DFT^1 method, only 1024 points (23 ms analysis time) were needed for the analysis window, and a great quality improvement was achieved. This result is represented at the bottom of Figure 9.

On most of high-pitched sounds (more than 180 Hz), excellent results have been achieved with very short analysis windows, down to 256 points (i.e. less than 6 ms analysis time).

7 Conclusion

In this article FT^n - n-order (short-time) Fourier Transform - has been introduced. This method is an enhancement of the standard short-time Fourier transform, providing greater accuracy for both frequency and amplitude with small analysis window, thus granting also great time resolution. This technique has been successfully tested on both synthetic and natural sounds with low noise. On the complexity point of view, this method is very interesting too, since it

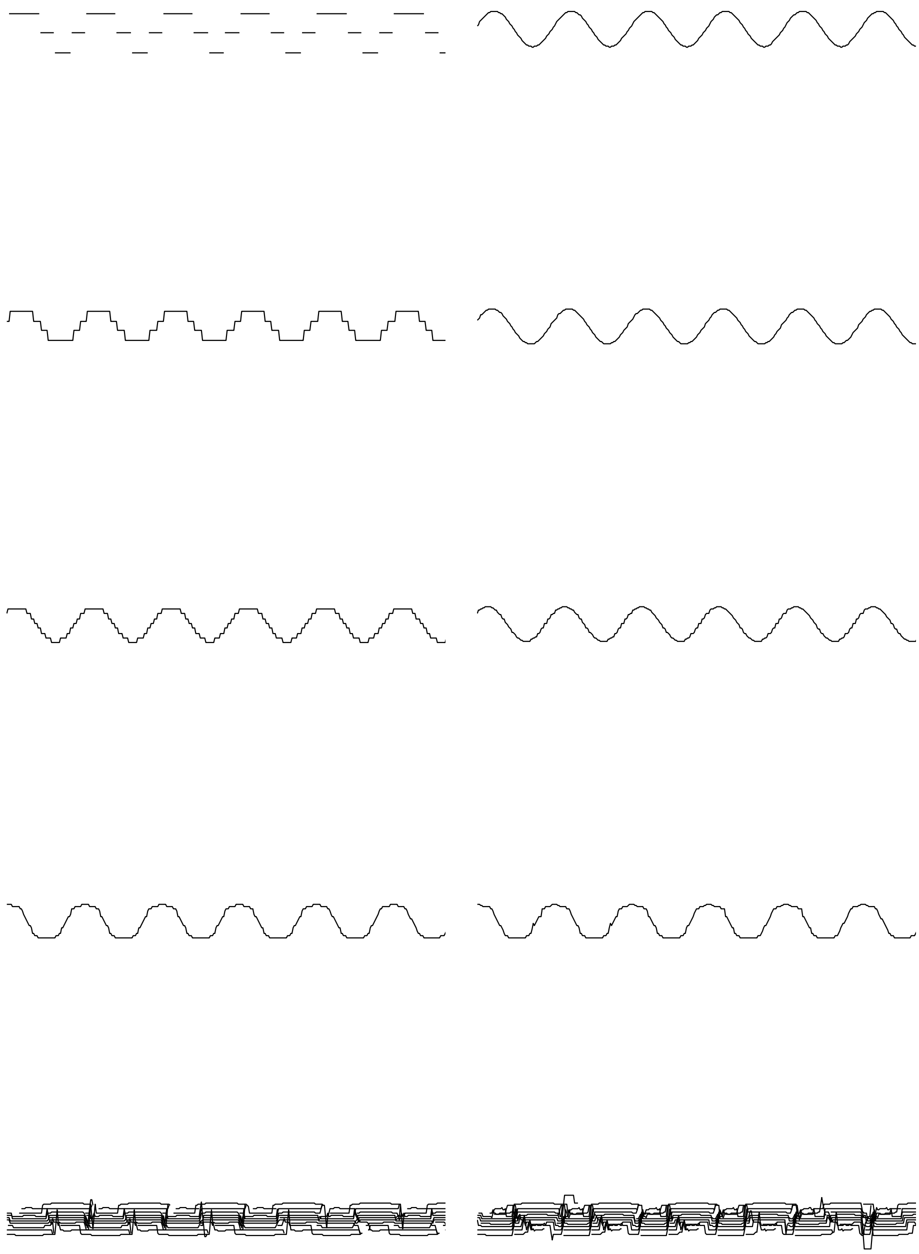

Figure 8: Comparison of DFT (left) and DFT^1 (right) on a single partial with vibrato. The analysis window width is increasing from top to bottom.

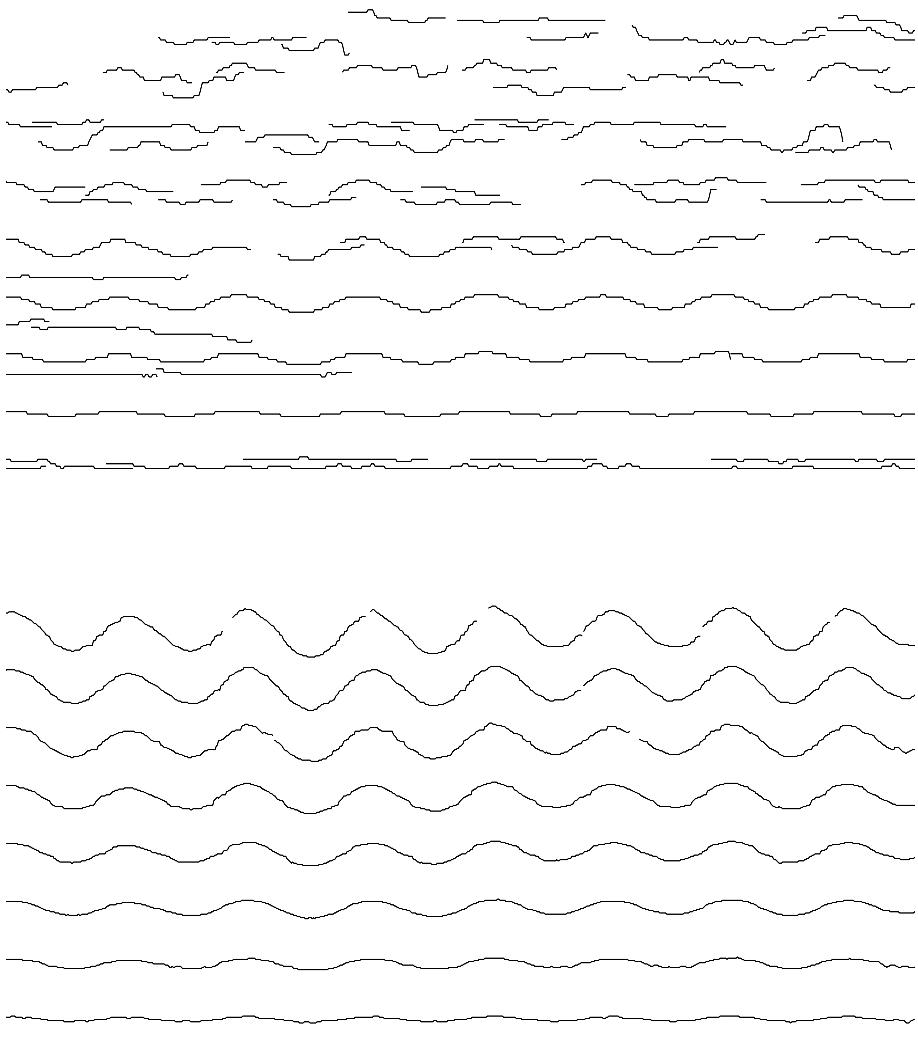

Figure 9: Voice with vibrato using standard DFT (top) and DFT^1 method (bottom).

requires the computation of two small discrete Fourier transforms instead of one much larger.

Originally designed for sounds with slow time-varying partials, this method has turned out to allow precise analysis of instruments with quite fast evolutions (even for the attack phase). This is indeed possible because small analysis window are sufficient for high-pitched sounds. The method can also be generalized for noisy sounds, which are fast time-varying, assuming that the stochastic part remains low. To do so, there is two possibilities: either revert to order 0, or continue to order 2 if computation precision makes it possible. In the first case, the spectral envelope of the noise can be easily found using the standard Fourier transform. In the second case, performing order 2 analysis might help precising the order 1 results to better isolate spectral peaks.

The method can practically be used during the analysis phase of a spectral modeling synthesis, instead of a classical phase vocoder. Designing a peak tracking algorithm taking advantage of the FT^n analysis should be of great interest.

This technique is quite general and may be used together with the wavelet transform instead of the short-time Fourier transform, as soon as the limitations and imprecisions of wavelet transform are clearly identified. However, the short-time Fourier transform with multiple analysis window size might lead to equivalent results.

References

[All77] J. B. Allen. Short-Term Spectral Analysis, Synthesis, and Modification by the Discrete Fourier Transform. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 25(3):235–238, 1977.

[Cas82] Michèle Castellengo. Sons Multiphoniques aux Instruments à Vent. Technical Report 34, IRCAM, 1982.

[Dol86] M. Dolson. The Phase Vocoder: A Tutorial. *Computer Music Journal*, 10(4):14–27, 1986.

[FH96] Kelly Fitz and Lippold Haken. Sinusoidal Modeling and Manipulation Using Lemur. *Computer Music Journal*, 20(4):44–59, 1996.

[Gre75] J. M. Grey. *An Exploration of Musical Timbre.* PhD thesis, Department of Psychology, Stanford University, 1975.

[Har78] F. J. Harris. On the use of windows for harmonic analysis with the discrete Fourier transform. In *Proceedings IEEE*, volume 66, pages 51–83, 1978.

[IRC96] IRCAM. *AudioSculpt user's manual*, second edition, April 1996.

[Moo77] J. A. Moorer. Signal Processing Aspects of Computer Music – A Survey. *Computer Music Journal*, 1(1):4–37, 1977.

[Moo78] J. A. Moorer. The Use of the Phase Vocoder in Computer Music Applications. *Journal of the Audio Engineering Society*, 26(1/2):42–45, 1978.

[Moo90] F. Richard Moore. *Elements of Computer Music.* Prentice-Hall, 1990.

[MP80] Max Mathews and John Pierce. Harmony and Nonharmonic Partials. Technical Report 28, IRCAM, 1980.

[MQ86] Robert J. McAulay and Thomas F. Quatieri. Speech Analysis/Synthesis Based on a Sinusoidal Representation. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 34(4):744–754, 1986.

[OS89] Alan V. Oppenheim and Ronald W. Schafer. *Discrete-Time Signal Processing.* Signal Processing Series. Prentice-Hall, 1989.

[Por80] M. R. Portnoff. Time-Frequency Representation of Digital Signals and Systems Based on Short-Time Fourier Transform. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 28(8):55–69, 1980.

[Ser97a] Marie-Hélène Serra. *Musical Signal Processing*, chapter Introducing the Phase Vocoder, pages 31–90. Studies on New Music Research. Swets & Zeitlinger, 1997.

[Ser97b] Xavier Serra. *Musical Signal Processing*, chapter Musical Sound Modeling with Sinusoids plus Noise, pages 91–122. Studies on New Music Research. Swets & Zeitlinger, 1997.

[SG84] J. O. Smith and P. Gosset. A Flexible Sampling-Rate Conversion Method. In *Proceedings IEEE ICASSP*, volume 2, pages 19.4.1–19.4.2, San Diego, March 1984.

[SS87] J. O. Smith and X. Serra. PARSHL: An Analysis/Synthesis Program for Non-Harmonic Sounds based on a Sinusoidal Representation. In *Proceedings of the 1987 International Computer Music Conference*, San Francisco, 1987. Computer Music Association.

[SS90] Xavier Serra and Julius O. Smith. Spectral Modeling Synthesis: A Sound Analysis/Synthesis System Based on a Deterministic plus Stochastic Decomposition. *Computer Music Journal*, 14(4):12–24, 1990.