Reducing Aliasing from Synthetic Audio Signals Using Polynomial Transition Regions

Jari Kleimola and Vesa Välimäki, *Senior Member, IEEE*

***Abstract*—Sampling of discontinuous audio signals with rich spectra is a valuable asset in subtractive synthesis, but results in aliasing distortion. This letter proposes an aliasing-reduction technique, which is cost-effective, transient-free, and extensible to various discontinuities. It replaces the samples on a finite region around each discontinuity with values taken from a smooth polynomial, based on a novel interpretation of the differentiated polynomial waveform (DPW) method. In the musical pitch range, the number of operations in the proposed method is at least 40% smaller than in the DPW algorithm. The method is widely usable in sound synthesis applications.**

***Index Terms*—Acoustic signal processing, antialiasing, audio oscillators, music, signal synthesis.**

I. Introduction

DIGITAL subtractive synthesis of audio signals requires a way of reducing aliasing from its input signal, which is usually a classical waveform [1], [2]. Straightforward sampling of such discontinuous waveforms leads to unpleasant aliasing distortion, unless the fundamental frequency is very low with respect to the sample rate. This letter proposes to reduce aliasing by replacing a few samples around each discontinuity with values taken from a smooth polynomial. The new method is cost-effective and can be applied to miscellaneous signals containing discontinuities.

The idea to implement alias-reduced waveforms for subtractive synthesis was proposed by Stilson and Smith, who suggested generating first an approximately bandlimited impulse train (BLIT) using a look-up table and then filtering the impulse train to obtain the desired waveshape [3]. This is generally more efficient and flexible than using additive synthesis [4], [5], which would otherwise be the method of choice, as it offers a completely alias-free solution. Brandt developed an improved approach, in which an approximately bandlimited step function (BLEP) is used for deriving a correction function for each discontinuity [6]. The BLEP correction function, or BLEP residual, is obtained by integrating the sinc function or the impulse response of a lowpass filter and by subtracting an ideal unit step function from it [2], [6].

Manuscript received September 26, 2011; revised November 09, 2011; accepted November 11, 2011. Date of publication November 30, 2011; date of current version December 19, 2011. This work was supported by the Academy of Finland under Project 122815. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Constantine L. Kotropoulos.

The authors are with Aalto University School of Electrical Engineering, Department of Signal Processing and Acoustics, FI-00076 Aalto, Espoo, Finland (e-mail: jari.kleimola@aalto.fi; vesa.valimaki@tkk.fi).

Digital Object Identifier 10.1109/LSP.2011.2177819

Nam and his co-workers have proposed an alternative BLIT method implemented using fractional delay filters (BLIT-FDF) [7]. In that method, an impulse response of a polynomial interpolation filter is generated once every period, thus avoiding the look-up table of the original BLIT algorithm. Similarly, the BLEP method has been converted into a polynomial algorithm, in which the BLEP residual function is derived using low-order polynomial functions (PolyBLEP) [2].

Välimäki has invented the simplest alias-reducing method to date, which is based on polynomial waveshaping, differentiating, and scaling a trivial sawtooth waveform, i.e., a bipolar modulo counter output [8]. Squaring the bipolar modulo counter output leads to a piecewise parabolic signal, which is equivalent to the integral of the sawtooth waveform, and the differentiator can be implemented using the first-order difference operator. This algorithm has been recently extended to an arbitrary number of integrations and differentiations, and is called the differentiated polynomial waveform (DPW) method [9]. The DPW method has a perceptual advantage, as aliasing is suppressed most in the lower half of the spectrum, where the hearing is most sensitive. The drawbacks of this approach are the cumbersome scaling of the output signal [8], [9] and the requirement for differentiable polynomial waveforms.

In this letter, we derive an alternative realization of the DPW algorithm and extend its applicability. This advancement is based on the observation that the DPW method [8], [9] only modifies samples of the modulo-counter signal on a finite region around each discontinuity and offsets the other samples, although all signal samples go through the same processing steps. The modified sample values are equal to those taken from a polynomial. As a result, we propose an alternative algorithm that shares the alias-reduction qualities of the DPW method without unnecessary operations. This novel approach, which we call PTR (polynomial transition regions), is cost-effective, and free of the transient problems we encountered in the DPW algorithm. Furthermore, it can be directly applied to any discontinuous waveform, not just the classical waveforms.

The letter is structured as follows. In Section II, the DPW algorithm is briefly summarized and reinterpreted. In Section III, the new polynomial transition region method is introduced and its computational load is compared to previous methods. Application of the PTR method to the alias-reduction of nontrivial waveforms is demonstrated in Section IV. Section V concludes this letter.

II. Time-Domain Properties of the DPW Algorithm

A. Summary of the DPW Algorithm

The DPW method [8], [9] produces alias-suppressed sawtooth waveforms in four stages. First, a unipolar modulo counter

signal $\phi(n)$ is transformed into a trivial bipolar sawtooth $s(n)$ using expressions [8], [9]

$$\phi(n) = nT_0 \bmod 1, \tag{1}$$
$$s(n) = 2\phi(n) - 1 \tag{2}$$

where n is the sample number, $T_0 = f_0/f_s$ is the phase increment, f_0 is the fundamental frequency, and f_s is the sampling rate. The trivial sawtooth signal $s(n)$ is then processed with an S_N-order polynomial waveshaper (using a successive integral sN as the transfer function, e.g., s^2, $s^3 - s$, or $s^4 - 2s^2$ [9]), and in the third stage, differentiated $N-1$ times to revert to the piecewise linear form. The result is finally multiplied by a scaling factor to restore the differentiated signal level. The alias-suppressed sawtooth waveform is thus given by [9]

$$y(n) = \frac{P_0^{N-1}\nabla^{N-1}s_N(n)}{2^{N-1}N!} \tag{3}$$

where ∇^{N-1} is the backward difference operator $\nabla s(n) = s(n) - s(n-1)$ applied $N-1$ times, $s_N(n)$ is the waveshaper polynomial, and $P_0 = 1/T_0$ is the period of the unipolar modulo counter $\phi(n)$. This process reduces aliasing because the spectra of the resulting signals decay about $6N$ dB per octave (i.e., as a function of the polynomial order), and then the differentiation stage restores the spectral tilt towards that of a sawtooth waveform. In the time domain, earlier work [2], [8]–[10] noted that for $N = 2$, the alias-suppressed waveform differs from the trivial waveform by a single sample and a shift. This provides a smoother transition from one period to another. However, previous work does not explicitly interpret the time-domain behavior of the DPW method.

B. Novel Interpretation of the DPW Algorithm

We noticed that the transition region between two successive alias-suppressed waveform periods starts at the instant of the modulo operation of (1), and that because of successive differentiations in (3), this region is $W = N - 1$ samples wide [see Fig. 1(a)]. The alias-suppressed output signal can therefore be sliced into

$$y(n) = \begin{cases} y_{\mathrm{A}}(n), & \text{when } \phi(n) < WT_0 \\ y_{\mathrm{B}}(n), & \text{when } \phi(n) \geq WT_0 \end{cases} \tag{4}$$

where the subscripts A and B denote the segments inside and outside the transition region, i.e., in Fig. 1(a) samples 9–11 and 12–26, respectively. The expressions of $y_{\mathrm{A}}(n)$ and $y_{\mathrm{B}}(n)$ are derived here by first expanding (3) for the case of $N = 2$, and then extending the results into higher DPW orders. For brevity of expression, we restrict $0 \leq n < 2P_0$, and note that the results are applicable to subsequent periods because of the modulo-operation of (1).

For $W = 1$ (i.e., $N = 2$), the waveshaper is $s_2(n) = s^2(n) = [2\phi(n) - 1]^2$ and $\nabla s_2(n) = s_2(n) - s_2(n-1)$. Outside the transition region, the modulo operation of (1) is inactive, and therefore $\phi(n) = nT_0$ and $\phi(n-1) = (n-1)T_0$. Substitution and straightforward development of (3) leads to

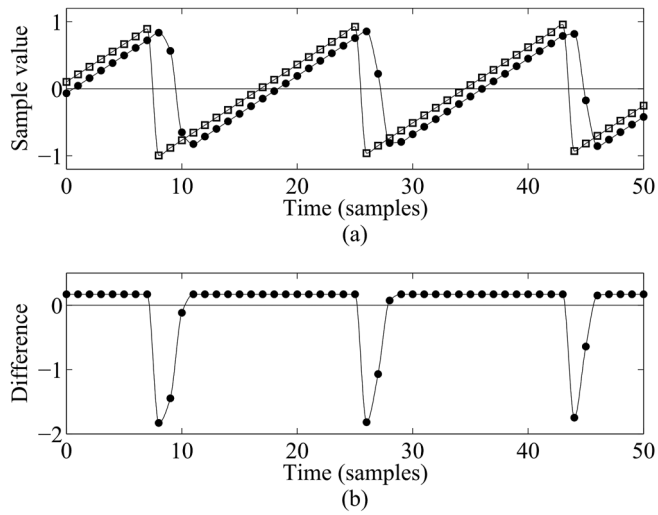

Fig. 1. (a) The trivial sawtooth (squares), the DPW sawtooth with $N = 4$ (dots), and (b) their difference ($f_0 = 2490$ Hz, and $f_s = 44.1$ kHz as in all plots in this letter). The modulo operation is activated at $n = 8$, 26, and 44.

$$y_{\mathrm{B},W=1}(n) = \frac{[2nT_0 - 1]^2 - [2(n-1)T_0 - 1]^2}{4T_0} = 2nT_0 - 1 - T_0. \tag{5}$$

Inside the initial transition region, the modulo operation of (1) is active, yielding $\phi(n) = nT_0 - 1$ and $\phi(n-1) = (n-1)T_0$. Applying (3) gives now

$$y_{\mathrm{A},W=1}(n) = 2nT_0 - 1 - T_0 - 2(n - P_0) + 2. \tag{6}$$

The first two terms of the right-hand side of (5) and (6) form $s(n) = 2nT_0 - 1$, i.e., the trivial bipolar sawtooth signal of (2), while the third term T_0 represents an offset. Within the initial transition, the $(n - P_0)$ term of (6) defines the distance from the discontinuity to the samples following it. This can be expressed in terms of $\phi(n)$ as $D(n) = \phi(n)P_0 = \phi(n)/T_0$, so that $0 \leq D(n) < W$. Comparing (5) and (6) reveals that $y_{\mathrm{A},W=1}(n) = y_{\mathrm{B},W=1}(n) + c_1[D(n)]$, where the correction polynomial $c_1[D(n)] = -2D(n) + 2$ is responsible for the transition smoothing between the waveform periods.

Repeating this derivation for higher DPW orders gives the general cases

$$y(n) = \begin{cases} s(n) - WT_0 + c_W\left[D(n)\right], & \text{when } \phi(n) < WT_0 \\ s(n) - WT_0, & \text{when } \phi(n) \geq WT_0 \end{cases} \tag{7}$$

where the WT_0 term corresponds to the dc offset [see Fig. 1(b)], and $c_W[D(n)]$ is the correction polynomial, as listed at the accompanying webpage of this letter [11].

To summarize, the novel interpretation of the DPW algorithm implies that DPW produces alias-suppressed sawtooth waveforms by generating an offset trivial sawtooth signal, which is adjusted with a correction polynomial when the phase counter is within the transition region. This leads to the definition of the new polynomial transition region algorithm.

TABLE I
PTR TRANSITION POLYNOMIALS p_W FOR AN ALIAS-SUPPRESSED SAWTOOTH SIGNAL AS FUNCTION OF DISTANCE FROM DISCONTINUITY $D(n) = \phi(n)P_0 = \phi(n)/T_0$

W	Span	Transition polynomial p_W
1	$0 \ldots T_0$	$(2T_0 - 2h)D + 2h - 1$
2	$0 \ldots T_0$	$-hD^2 + 2T_0D + 2h - 1$
	$T_0 \ldots 2T_0$	$hD^2 + (2T_0 - 4h)D + 4h - 1$
3	$0 \ldots T_0$	$-hD^3/3 + 2T_0D + 2h - 1$
	$T_0 \ldots 2T_0$	$2hD^3/3 - 3hD^2 + (2T_0 + 3h)D + h - 1$
	$2T_0 \ldots 3T_0$	$-hD^3/3 + 3hD^2 + (2T_0 - 9h)D + 9h - 1$

III. POLYNOMIAL TRANSITION REGION ALGORITHM

The polynomial transition region (PTR) algorithm is defined as

$$y(n) = \begin{cases} p_W(n) - c_{\mathrm{dc}}, & \text{when } \phi(n) < WT_0 \\ x(n) - c_{\mathrm{dc}}, & \text{when } \phi(n) \geq WT_0. \end{cases} \quad (8)$$

where $p_W(n)$ is the transition polynomial that replaces the input signal samples when the phase counter $\phi(n)$ is within the transition region that is W samples wide, $x(n)$ is the aliasing-contaminated input signal, and c_{dc} is the offset.

To explore the properties of the proposed algorithm, consider an efficient alias-suppressed sawtooth synthesis method based on the DPW reinterpretation of Section II-B. The aliasing-contaminated input signal $x(n) = 2\phi(n) - 1$ is offset by $c_{\mathrm{dc}} = WT_0$. The transition polynomials $p_W(n) = x(n) - c_{\mathrm{dc}} + hc_W[D(n)]$ are formed by adjusting the offset trivial signal with a scaled correction polynomial $hc_W[D(n)]$, where $h = 1$ for downward and $h = -1$ for upward transitions (see Table I).

The PTR form has an immediate advantage over the DPW method: PTR provides substantial savings in the computational cost since outside the transition region, which is in most cases considerably wider than $W = N-1$ samples, DPW reduces into a bipolar modulo counter with an offset. This is demonstrated by Fig. 1(b), which remains at a constant value outside the transition region. Yet, the DPW method applies (3) to every sample. Table II compares the computational load of the PTR method to previous DPW, B-spline BLIT-FDF [7] and PolyBLEP [2] methods in terms of number of operations per output sample, using (1) as the input signal. The DPW implementation is based on (3), while the PTR method computes (8) as explained earlier in this section. Horner's rule was applied for optimization (the source code is available at [11]).

Fig. 2 compares visually the computational load of DPW, PolyBLEP, BLIT-FDF, and PTR methods showing that the proposed PTR method has the smallest number of operations in all cases. Because of the branch operations in (8), certain computations are performed only a few times per waveform period, and therefore, the computational load of PTR depends on the fundamental frequency: since the length of the waveform period shrinks when the fundamental frequency rises, while the transition region width W remains constant, the relative savings decrease with increasing fundamental frequency.

However, PTR uses considerably less operations than the DPW method over the entire frequency range of interest, as shown in Fig. 2. At 4186 Hz, which is the fundamental frequency of the highest key of an 88-key piano keyboard, PTR

TABLE II
COMPUTATIONAL LOAD AS NUMBER OF OPERATIONS PER OUTPUT SAMPLE, OPTIMIZED FOR UNIPOLAR MODULO COUNTER INPUT

Method	Mul	Add	Branch	Total
$W = 1$				
DPW	3	2	–	5
BLIT-FDF	$1+2T_0$	$1+2T_0$	$1+T_0$	$3+5T_0$
PTR	**1**	**1**	**1**	**3**
$W = 2$				
DPW	4	4	–	8
PolyBLEP	$1+6T_0$	$1+7T_0$	$1+T_0$	$3+14T_0$
BLIT-FDF	$1+4T_0$	$1+5T_0$	$1+2T_0$	$3+11T_0$
PTR	$\mathbf{1+4T_0}$	$\mathbf{1+2T_0}$	$\mathbf{1+T_0}$	$\mathbf{3+7T_0}$
$W = 3$				
DPW	4	5	–	9
BLIT-FDF	$1+10T_0$	$1+8T_0$	$1+3T_0$	$3+21T_0$
PTR	$\mathbf{1+9T_0}$	$\mathbf{1+5T_0}$	$\mathbf{1+2T_0}$	$\mathbf{3+16T_0}$

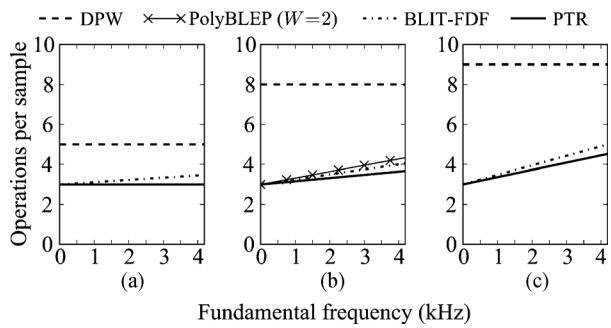

Fig. 2. Computational load of DPW (dashed), PolyBLEP (crosses), BLIT-FDF (dash-dotted) and the proposed PTR (solid lines) sawtooth signals over the fundamental frequency range of an 88-key piano keyboard: (a) $W = 1$, (b) $W = 2$, and (c) $W = 3$.

consumes 40%, 54%, and 50% less operations than DPW, for $W = 1$, 2 and 3, respectively (see Fig. 2). We found the savings to be smaller in practical implementations, but still favorable with PTR. In a benchmark test involving a general purpose CPU with 20 banks of 88-voice sawtooth oscillators, PTR $(W = 3)$ used 27% of the available CPU power, in comparison to the 36% consumed by the DPW method. It can be deduced from (8) and Table I that, for $h = 1$, the overhead of the transition polynomial coefficient calculation per fundamental frequency change is 4, 6, and 9 operations for $W = 1$, 2 and 3, respectively, which is acceptable for control rate updates.

Furthermore, it has been noticed that the differentiator state variables of (3) generate audible transients when f_0 is changed rapidly in the DPW method. PTR is state-free, and does not suffer from transient problems. This additional advantage is illustrated in Fig. 3, which shows that the transient present in the DPW implementation is not reproduced in the PTR form. BLIT-FDF and PolyBLEP are likewise transient-free, although the leaky integration in the former approach causes dc fluctuations.

IV. EXTENSION TO OTHER DISCONTINUITIES

The third advantage of the PTR algorithm is that it scales to arbitrary transition heights, and that it can be applied to other waveforms besides the trivial sawtooth—even to those that are nondifferentiable. As an example, this section explores the

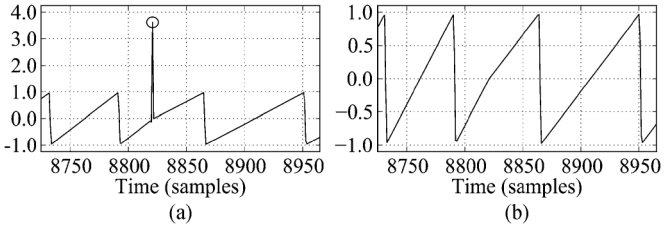

Fig. 3. Ramp-like frequency modulation $f_{\mathrm{m}} = 10$ Hz from $f_0 = 500$ Hz to $f_0 = 750$ Hz using (a) DPW and (b) PTR ($W = 2$). The disturbing transient in (a) is indicated with a circle.

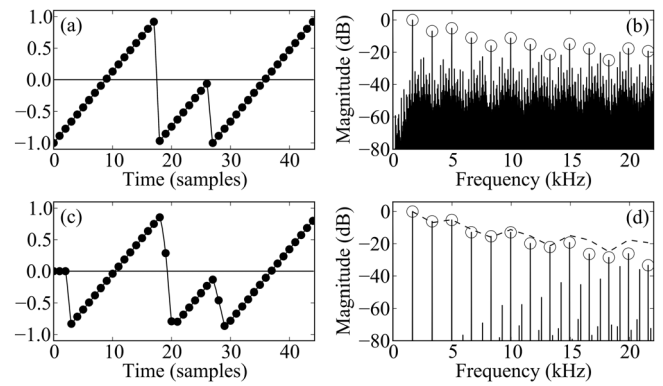

Fig. 4. Waveform and spectrum of (a), (b) trivial, and (c), (d) PTR oscillator hard synchronization algorithms ($f_{\mathrm{master}} = 1660$ Hz, $r = 1.5$, $W = 3$). The desired harmonics in (b) and (d) are indicated with circles, and the dashed line in (d) plots the harmonic envelope of the trivial spectrum.

alias-suppressed oscillator hard synchronization effect, whose implementation is impractical using the DPW method.

In classic oscillator hard synchronization (hard sync) [6], the phase of the slave oscillator, which generates the output signal, is reset each time the master oscillator completes its cycle. The frequency of the master oscillator f_{master} determines the fundamental frequency of the output, while the frequency of the slave f_{slave} controls the timbre of the produced sound. The produced timbre is rich and complex, and because the phase resets generate abrupt discontinuities in the output, straightforward digital hard sync implementations suffer from excessive aliasing, as shown in Fig. 4(a) and (b). The desired harmonics in Fig. 4(b) are indicated with circles—the rest of the spectral components are aliasing.

The DPW method fails to reduce the aliasing of hard sync implementations, because the derivatives of the waveshapers $s_N(n)$ of (3) are discontinuous at the instant of phase reset. This is not of concern in PTR, which applies the transition polynomials of Table I separately for each discontinuity. However, the correction term of the transition polynomial has to be scaled according to the height h of the discontinuity at the instant of the reset, which is determined by the frequency ratio $r = f_{\mathrm{slave}}/f_{\mathrm{master}}$ as

$$h = \begin{cases} 1, & \text{when } r \text{ is an integer} \\ r - \mathrm{floor}(r), & \text{otherwise.} \end{cases} \tag{9}$$

To preserve the phase continuity at the fundamental frequency, the phase of the slave oscillator is reset to $\phi_{\mathrm{s}} = r\phi_{\mathrm{m}}$, where ϕ_{m} is the phase of the master oscillator after the modulo operation. Figs. 4(c) and 4(d) show that the aliasing present in the trivial hard sync spectrum is reduced substantially when exploiting the PTR algorithm. The high end of the spectrum is slightly attenuated, which is typical to alias-suppression methods [9], but this can be compensated with a low-order equalizer, if desired. Audio examples and further applications are available at [11].

V. Conclusions

This letter provided a time-domain reinterpretation of the DPW method, which lead to the definition of a novel alias-suppression algorithm called Polynomial Transition Regions (PTR). The proposed method replaces the discontinuities of the aliasing-contaminated input signal with polynomial functions, while offsetting the other samples. The method was shown to have several advantages over the DPW method, including computational efficiency, transient-free operation, and applicability to various waveforms having discontinuities. The computational cost of PTR compares favorably against previous BLIT-FDF and PolyBLEP methods. Furthermore, since the polynomial, which defines the transition region, does not have to be differentiable, the PTR algorithm enables experimenting with and optimizing of alternative polynomial functions. This is left for future work.

Acknowledgment

Dr. N. Collins is acknowledged for posing questions on transient problems of the DPW algorithm.

References

[1] T. Stilson, "Efficiently-Variable Non-Oversampling Algorithms in Virtual-Analog Music Synthesis—A Root-Locus Perspective," Ph.D. dissertation, Dept. Elect. Eng, Stanford Univ., Stanford, CA, June 2006.

[2] V. Välimäki and A. Huovilainen, "Antialiasing oscillators in subtractive synthesis," *IEEE Signal Process. Mag.*, vol. 24, no. 2, pp. 116–125, Mar. 2007.

[3] T. Stilson and J. Smith, "Alias-free digital synthesis of classic analog waveforms," in *Proc. Int. Computer Music Conf.*, Hong Kong, Aug. 1996, pp. 332–335.

[4] G. Winham and K. Steiglitz, "Input generators for digital sound synthesis," *J. Acoust. Soc. Amer.*, vol. 47, pt. 2, pp. 665–666, Feb. 1970.

[5] J. A. Moorer, "The synthesis of complex audio spectra by means of discrete summation formulae," *J. Audio Eng. Soc.*, vol. 24, no. 9, pp. 717–727, Nov. 1976.

[6] E. Brandt, "Hard sync without aliasing," in *Proc. Int. Computer Music Conf.*, Havana, Cuba, Sep. 2001, pp. 365–368.

[7] J. Nam, V. Välimäki, J. S. Abel, and J. O. Smith, "Efficient antialiasing oscillator algorithms using low-order fractional delay filters," *IEEE Trans. Audio, Speech Lang. Process.*, vol. 18, no. 4, pp. 773–785, May 2010.

[8] V. Välimäki, "Discrete-time synthesis of the sawtooth waveform with reduced aliasing," *IEEE Signal Process. Lett.*, vol. 12, no. 3, pp. 214–217, Mar. 2005.

[9] V. Välimäki, J. Nam, J. O. Smith, and J. S. Abel, "Alias-suppressed oscillators based on differentiated polynomial waveforms," *IEEE Trans. Audio, Speech Lang. Process*, vol. 18, no. 4, pp. 786–798, May 2010.

[10] V. Välimäki and A. Huovilainen, "Oscillator and filter algorithms for virtual analog synthesis," *Comput. Music J.*, vol. 30, no. 2, pp. 19–31, 2006, Summer.

[11] J. Kleimola and V. Välimäki, Polynomial Transition Regions [Online]. Available: http://www.acoustics.hut.fi/go/spl-ptr Nov. 9, 2011, [Nov. 21, 2011].