Note on Alias Suppression in Digital Distortion

Martin Vicanek

4. November 2023, revised 19. September 2024

Abstract

Various methods and their interplay for alias suppression in digital distortion are considered, some of them refined.

1 Introduction

Aliasing is a notorious problem in digital sound synthesis and processing. In distortion units, high-frequency components occur as a result of a nonlinear transfer function. Applied point-wise in the digital domain, this amounts to direct sampling of the distorted signal. Frequencies above Nyquist will fold back and produce unwanted anharmonic noise.

Oversampling is an obvious and often used technique to reduce aliasing. An alternative approach published recently is based on convolution (or, more generally, lowpass filtering) in the continuous-time domain [1]. The authors propose, in lowest order, a rectangular filter kernel. If the width is taken equal to the sample separation, then the filter transfer function, which is $\mathrm{sinc}(\pi f/f_s)$, where f_s is the sampling frequency, has zeros at multiples of the sampling frequency. This is a very desirable property, as these frequencies get folded back to zero frequency in the digital domain. Hence alias suppression will be most effective in the (digital) low frequency region.

Consider a digital input sequence x_n sent through a naive wave shaper, so that the output sequence is

$$y_n = f(x_n) \tag{1}$$

I call the wave shaper in eq.(2) *naive* because it will produce unwanted aliasing, making it of limited use.

The authors of [1] suggest to convert the sequence x_n first into the continuous domain via linear interpolation, then apply the shaper, and subsequently perform a convolution with a rectangular kernel. Sampling the result to get back to the digital domain, they arrive at the following formula:

$$y_n = \frac{F(x_n) - F(x_{n-1})}{x_n - x_{n-1}} \tag{2}$$

where $F(x)$ is the antiderivative of $f(x)$. They provide examples for $f(x) = \tanh(x)$ and for a hard clipper. They point out that numerical evaluation is ill-conditioned for x_n close to x_{n-1}.

2 Making it transparent

Obviously eq.(2) introduces a latency of one half sample. This may not be a concern for many applications, however, there is also some lowpassing happening. For a transparent shaper, $f(x) = x$, eq.(2) yields

$$y_n = \frac{x_n + x_{n-1}}{2}, \tag{3}$$

which means that the highest frequencies will be lost (the transfer function has a zero at Nyquist frequency, refer to the blue curve in figure 1). This may be perceived as a flaw: a transparent shaper should not remove part of the spectrum!

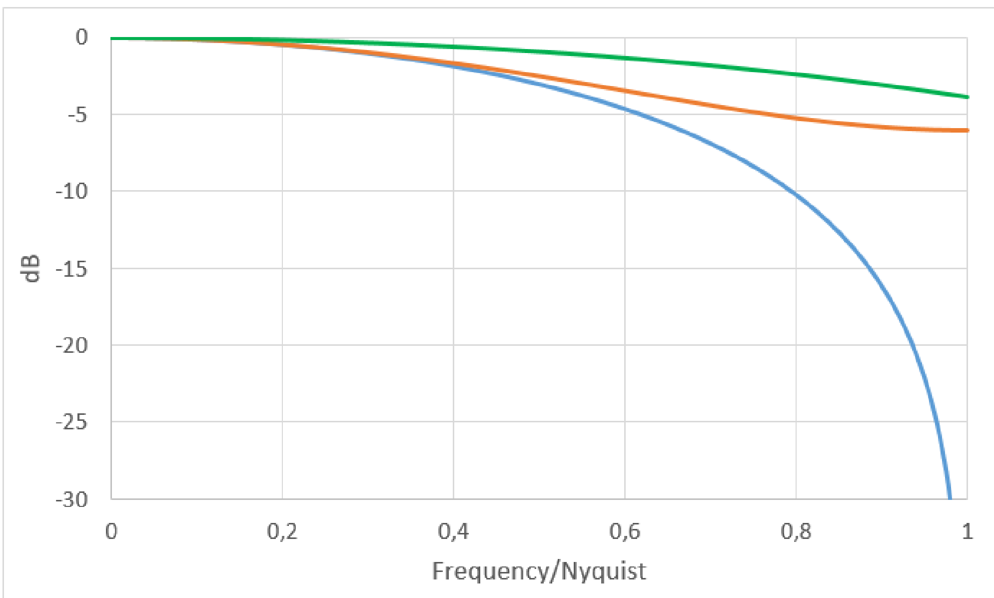

Figure 1: Various lowpass filter responses. Blue: eq.(3). Red: eq.(5). Green: convolution with a rectangular kernel of 1 sample width.

This shortcoming may be mitigated by a slight variation of the scheme presented in [1]. Delaying the convolution kernel by half a sample, we arrive at a relation similar to eq.(2),

$$y_n = \frac{F_{1/2} - F_1}{x_n - x_{n-1}} + \frac{F_1 - F_{3/2}}{x_{n-1} - x_{n-2}} \tag{4}$$

where $F_{1/2}$ denotes $F(\frac{x_n + x_{n-1}}{2})$, $F_1 = F(x_{n-1})$, and $F_{3/2}$ stands for $F(\frac{x_{n-1} + x_{n-2}}{2})$. Eq.(4) is slightly more complex than eq.(2) but reduces aliasing equally well. It has a similar issue when consecutive samples are nearly equal, $x_n \approx x_{n-1}$ or $x_{n-1} \approx x_{n-2}$, resulting in loss of precision. Whenever this happens, replace the corresponding fraction in eq.(4) by $\frac{1}{2} f(x_{n-1})$. How close exactly should the samples be for the replacement to be an advantage? That depends on the nature of the shaping function $f(x)$ in the relevant region, and how accurately one can compute the antiderivative $F(x)$. As a rule of thumb, if ϵ is the machine precision, a threshold $\sqrt{\epsilon}$ is not a bad choice.

Eq.(4) introduces one sample delay, however the lowpass effect is more gentle. For a transparent shaper, eq.(4) becomes

$$y_n = \frac{x_n + 6x_{n-1} + x_{n-2}}{8}. \tag{5}$$

The attenuation at Nyquist frequency is only 6 dB (orange curve in figure 1).

Since eq.(5) describes a filter with two zeros on the z-plane, $z_1 = -1/(3+\sqrt{8})$ and $z_2 = 1/z_1$, the magnitude response may be easily equalized by adding an IIR-filter with poles determined by the zero locations z_1 and z_2. One pole will be at z_1. Because z_2 lies outside the unit circle, we place the corresponding pole at its mirror image (which happens to be z_1) to obtain a stable IIR-filter. This will leave the magnitude response unaffected. Hence we have two similar one-pole filters in series,

$$y_n = bx_n - ay_{n-1} \tag{6}$$

with $a = 1/(3+\sqrt{8})$ and $b = 1 + a$. We suggest to place one of the filters before the shaper, and the other after the shaper to make up for the lowpass effect of the convolution (sinc filter). The latter reduces the shaped signal by about 4 dB at Nyquist (green curve in figure 1).

With the alternative eq.(4) and the compensation 1-pole filters in eq.(6) we have created an alias suppression scheme which is perfectly transparent in the linear regime.

3 Making it simple

A particularly simple expression is obtained for the shaper function

$$f(x) = \frac{x}{\sqrt{1+x^2}}, \tag{7}$$

for which the antiderivative is

$$F(x) = \sqrt{1+x^2} \tag{8}$$

Then eq.(2) may be cast into the form

$$y_n = \frac{\sqrt{1+x_n^2} - \sqrt{1+x_{n-1}^2}}{x_n - x_{n-1}} = \frac{x_n + x_{n-1}}{\sqrt{1+x_n^2} + \sqrt{1+x_{n-1}^2}} \tag{9}$$

which allows numerically stable evaluation for all x_n and x_{n-1}. Similarly, eq.(4) may be written as

$$y_n = \frac{1}{4}\left(\frac{x_n + 3x_{n-1}}{F_{1/2} + F_1} + \frac{3x_{n-1} + x_{n-2}}{F_1 + F_{3/2}}\right) \tag{10}$$

with

$$F_{1/2} = \sqrt{1 + (\tfrac{x_n + x_{n-1}}{2})^2}$$
$$F_1 = \sqrt{1 + x_{n-1}^2}$$
$$F_{3/2} = \sqrt{1 + (\tfrac{x_{n-1} + x_{n-2}}{2})^2}.$$

The shaper function in eq.(7) is very similar to the popular $\tanh(x)$ function. The latter, however, is computationally more demanding, and furthermore requires special care in the numerical evaluation of eq.(2) when x_n is close to x_{n-1}. A detailed analysis of the harmonic intensities [2] revealed no significant differences between the two functions, hence we prefer to use the shaper function in eq.(7) rather than $\tanh(x)$.

4 ADAA versus oversampling

So how does the outlined scheme based on continuous-time domain filtering (in the following I will use the established term *antiderivative antialiasing*, ADAA for short) perform with regard to alias suppression compared to oversampling? Figure 2 illustrates the situation.

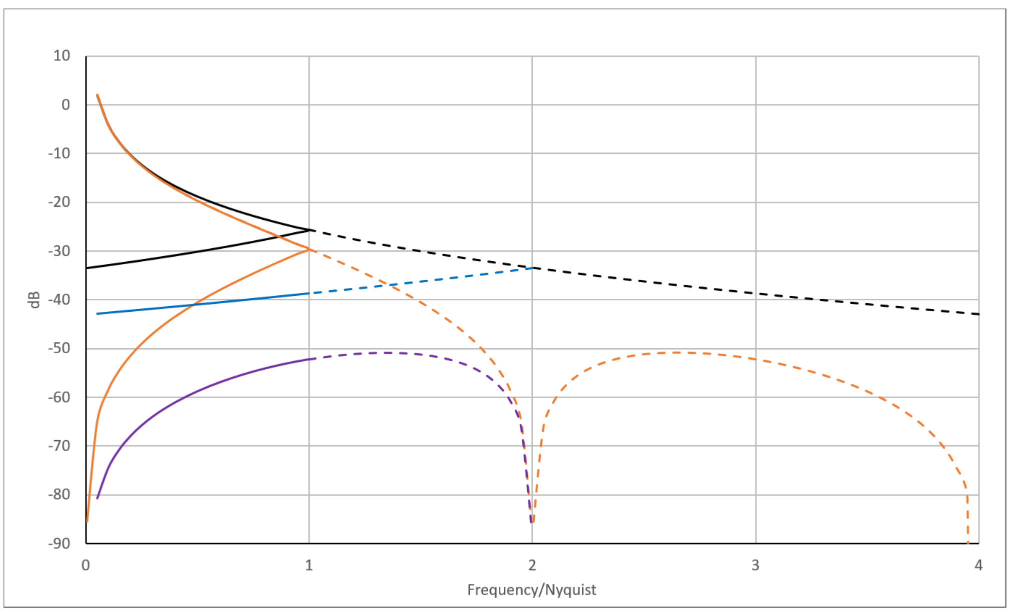

Figure 2: Aliasing and alias suppression. Black: spectrum of a hard-driven saturator, plotted as a continuous line for better visibility. The dashed line marks the part beyond Nyquist frequency, which folds back into the audible band as an alias. Subsequent foldings, which lead to more aliases, are left out for clarity. Blue: 2x oversampling. Orange: ADAA. Magenta: ADAA and 2x oversampling.

We already noted that ADAA is good at reducing low-frequency aliases, which is desirable (orange curve in figure 2). However, it is less efficient in the high frequency range, simply because the sinc filter is not very steep.

Alias reduction by oversampling depends on how far the signal reaches out beyond Nyquist. The steeper the falloff, the more efficient is oversampling. Conversely, for a spectrum with a mild falloff $\propto 1/f$, as is the case for instance for a hard driven saturator, 2x oversampling only reduces aliasing by 6 dB, 4x oversampling by 12 dB, and so on (blue curve in figure 2).

In fact ADAA and oversampling work best together (magenta curve in figure 2): ADAA makes the falloff steep so that oversampling can shine. ADAA combined with 4x oversampling makes aliasing practically a non-issue even for a hard clipper.

ADAA can be taken to higher orders in various ways [3]. Naturally, this results in more complex expressions, and may even introduce additional zeros in the Nyquist band [4], which in turn requires more oversampling to mitigate. I have not pursued this possibility because I am quite happy with first order ADAA and oversampling, however it may be worth the extra effort in applications with higher demands on aliasing suppression.

I am grateful to the members of The Audio Programmer Discord server for interesting discussions and valuable comments.

References

[1] Julian D. Parker, Vadim Zavalishin, Efflam Le Bivic, Reducing the Aliasing ofNonlinear Waveshaping Using Continuous-Time Convolution, Proceedings of the 19th International Conference on Digital Audio Effects (DAFx-16), Brno, Czech Republic, September 5–9, 2016

[2] M. Vicanek, Waveshaper Harmonics, 2021, https://vicanek.de/articles/WaveshaperHarmonix.pdf

[3] Stefan Bilbao, Fabián Esqueda, Julian D. Parker, and Vesa Välimäki, Antiderivative Antialiasing for Memoryless Nonlinearities, IEEE Signal Processing Letters, Vol. X, No. X, Nov. 2016

[4] Stefan Bilbao, Fabián Esqueda, and Vesa Välimäki, Antiderivative Antialiasing, Lagrange Interpolation and Spectral Flatness, 2017 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics, October 15-18, 2017, New Paltz, NY