Digital Signal Processing with a Focus on Transformation of Frequency Ranges

Hendrik Hangmann
University of Paderborn
h.hangmann@mail.uni-paderborn.de

December, 22 2013

Abstract

Digital signal processing is about the signal manipulation in digital systems. The goal of DSP is usually to measure, filter and compress continuous real-world analog signals. Areas of application are nowadays mainly the manipulation or compression of movie and sound. Especially in the field of sound the main task of DSP is to filter audio frequencies. This is done by two kinds of filters: the finite impulse response (FIR) and infinite impulse response (IIR). These filters enable the common bandpass, low-pass and high-pass filters. This paper gives an overview of the realization of these filters, especially the transformation of frequency ranges.

1 Introduction

1.1 Analog and Digital Signal Processing

Digital Signal Processing is the digital manipulation of analog signals. Basically the processing on these digitized signals is done on dedicated processors in real-time. The execution on general purpose processors is also possible, though considerably slower than on Digital Signal Processors (DSPs). These DSPs operate on a series of discrete quantities or values. In contrary to that, Analog Signal Processing (ASP) is mathematically done on a set of continuous values, such as voltage, current or electric charge. Figure 1 depicts such an analog signal processing device, which works on analog values and manipulates them by differential equations.

On the other hand, Digital Signal Processing requires to digitize the analog data in the first place. Figure 2 shows the corresponding Digital Signal Processor. First, the continuous, analog values get digitized by Analog-Digital Converters (ADC), which are sampling the analog signal. After operating on the data's digital representation, the analog signal has to be recreated from the samples by using Digital-Analog Converters (DAC).

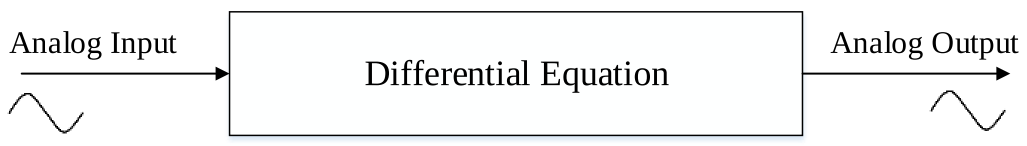

Figure 1: Analog Signal Processing cf. [3]

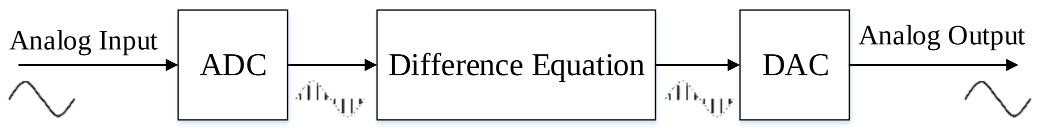

Figure 2: Digital Signal Processing cf. [3]

In contrary to ASP, which is using differential equations, DSP is working with difference equations, which will be covered later. The field of application range from military use (e. g. radar or sonar) to entertainment electronics, such as video, image or audio processing [5]. Especially in the latter, compressing and manipulation is done by ASP/DSP, more precisely by Filters, which can be both analog and digital. The main advantage of DSP and digital filters over ASP and analog filters is the linear phase response and the arbitrarily high filter order [3].

1.2 Structure

Chapter 2 is about filters in general. That means, the basics of filtering in time and frequency domain are discussed, as well as the transformations between them. Moreover the different possible filter functions are explained. Afterwards, in Chapter 3 the characteristics of digital filters, and particularly the Finite Impulse Response and Infinite Impulse Response Filters are presented and compared. Eventually, this paper is concluded in Chapter 4.

2 Filters

Basically, any medium which passes signals could be considered to be a filter, even the speakers of an audio system [4]. But usually we refer to filters as DSPs which manipulate signals frequency components, i. e. the amplitudes of the sinusoids of which the signal is composed. For instance an amplifier manipulates every frequency component and hence manipulates the complete signal's amplitude. However, filters are widely used to suppress, amplify or phase shift single frequencies or frequency ranges.

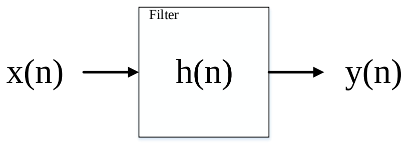

Figure 3: A basic filter with impulse response $h(n)$

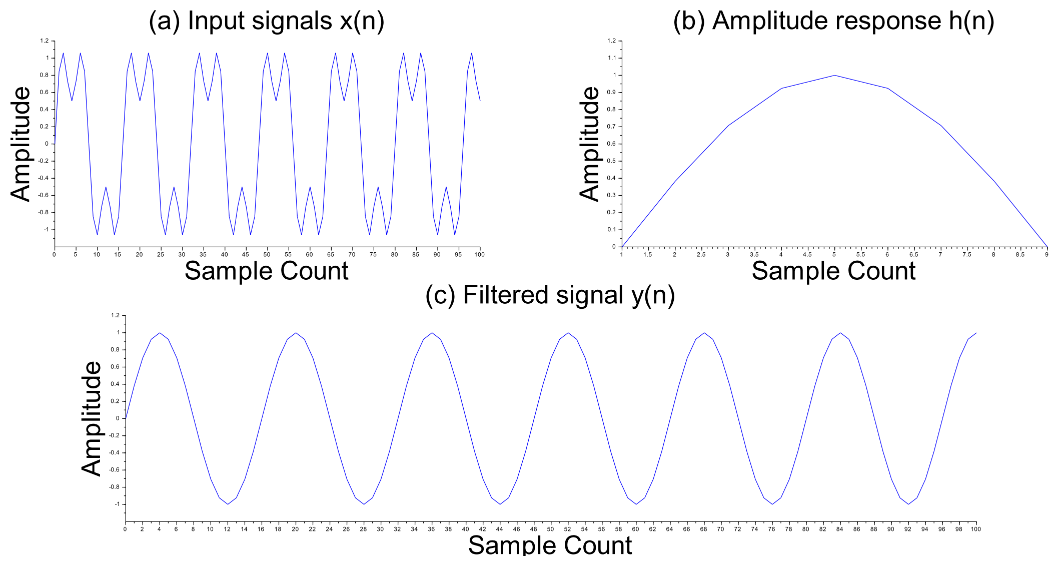

Figure 4: Filtering in Time Domain. (a) Input Signals $x(n)$ (b) Input Signals $x(n)$ (c) Filtered Signal $y(n)$

2.1 Filtering in Time Domain

Signals that use time (or sample count) as independent variable are said to be in the time domain [5]. I. e. each time step is assigned to an amplitude. This is the usual way to illustrate signals.

Filters in time domain work similar to the DSPs described in Section 1. Figure 3 depicts the basic idea of the functionality of a filter. It is fed by input signals $x(n)$ with an amplitude x at time n. These amplitudes are modified by the filter's impulse response $h(n)$ and finally output as filtered signal $y(n)$. This modification is performed by convolution with the impulse response $h(n)$. For complex values it is defined as follows.

$$y(n) = (x * h)(n) = \int_0^n x(\tau)h(n-\tau)\,\mathrm{d}\tau \tag{1}$$

However, DSP is mostly working on discrete functions. Hence filters are using the discrete convolution, which is defined as follows.

$$y(n) = (h * x)(n) = \sum_{m=0}^{n} x(m)h(n-m) \tag{2}$$

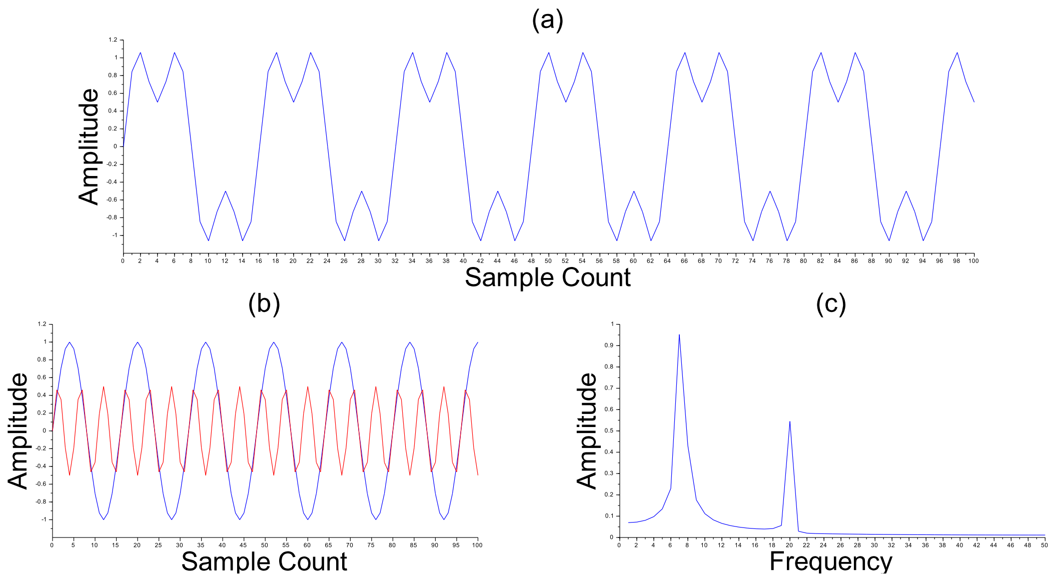

Figure 5: Example of the transformation to the frequency domain. (a) Input Signals $x(n)$ (b) Decomposed Input Signals of $x(n)$ (c) $X(\tilde{n})$: Spectrum of $x(n)$

Figure 4 illustrates such a filtering in the time domain. The input signal $x(n)$ in Figure 4a is convolved with the impulse response $h(n)$ in Figure 4b. The result of the convolution is then depicted in Figure 4c. Besides the impulse response, each filter also has a phase response, since the input phase might be delayed.

2.2 Filtering in Frequency Domain

A different way to display signals is in their frequency domain. This means the set of the sinusoid's amplitudes, that composes the wave in the time domain. The independent variable of the signal is hence the frequency [5]. I. e. each frequency is assigned to an amplitude. The frequency domain (or spectrum) is made up of each frequency, inclosed in the composed signal, scaled to the sampling rate.

For instance, the signal in Figure 5a is composed of the two sinusoids depicted in Figure 5b. The red sinusoid has a frequency three times higher than the blue one and half its amplitude. Transferring the input signal to the frequency domain yield an output like Figure 5c, which features two spikes. Each spike stands for the frequencies shown in Figure 5b.

The main idea behind filtering in the frequency domain is the convolution theorem, which states that convolution in the time domain is equal to multiplication in the frequency domain [1] as depicted in Equation 3, where $X(f)$ and $H(f)$ are transformed signals from the time domain. With the help of an efficient transformation, the costs of the multiplication in the frequency domain can be less than the convolution in the time domain.

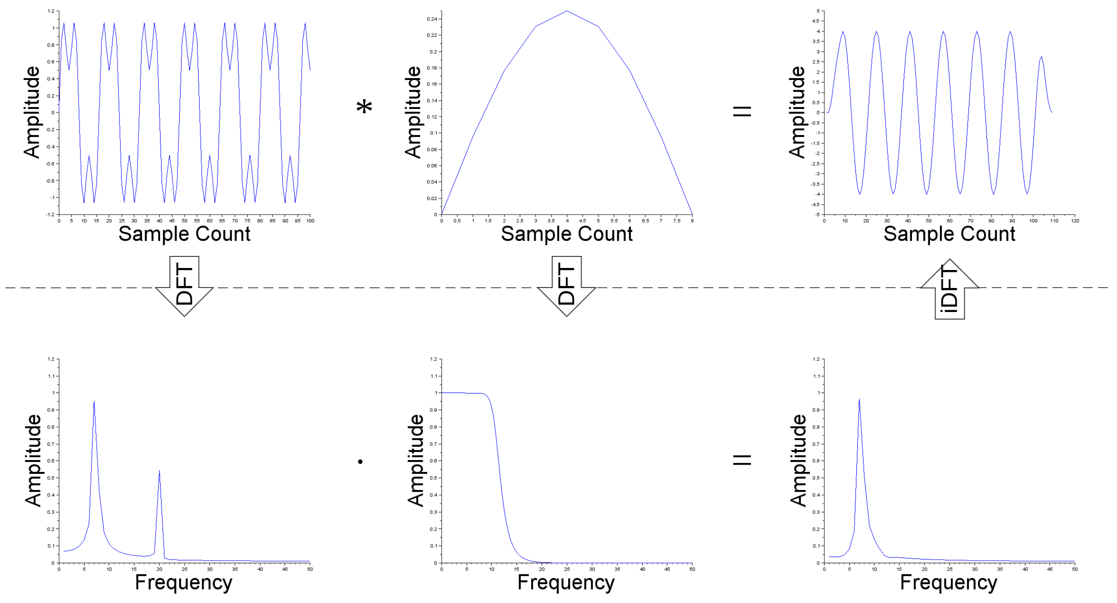

Figure 6: Graphical convolution theorem

$$(x * h)(n) \Rightarrow X(\tilde{n}) \cdot H(\tilde{n}) \tag{3}$$

Figure 6 illustrates the convolution theorem. The upper half depicts the time domain, where the signal is filtered with the help of convolution. The lower half in contrast shows the frequency domain where the transformed signals are simply multiplied frequency-wise. Note, that after multiplying the transformed signals they need to be retransformed to the time domain in order to feed them into a DAC. This transform (and re-transformation respectively) can be done with the help of the Discrete Fourier Transform (DFT) (and its inverse (iDFT) respectively). The transformation of the impulse response is also called the frequency response.

In practice, the multiplication paired with the DFTs is not faster then the calculation of the convolution. However, a variation of the DFT, the Fast Fourier Transform (FFT) is faster and is used.

2.3 Filter Functions

In the following, the basic filter functions are discussed. The representation of these functions is in frequency domain.

There are several ways to filter signals. One of them is the low-pass filter, as can be seen in Figure 7. This low-pass filter simply cuts out high frequencies beyond a frequency threshold (f_{stop}). Every frequency up to f_{pass} will be passed. $\frac{f_{stop}-f_{pass}}{2}$ is called cut-off frequency. The filter's maximum frequency is usually $0.5 \times f_a$, where f_a is the sampling rate. This is because of the Nyquist-Shannon Sampling Theorem, which indicates that

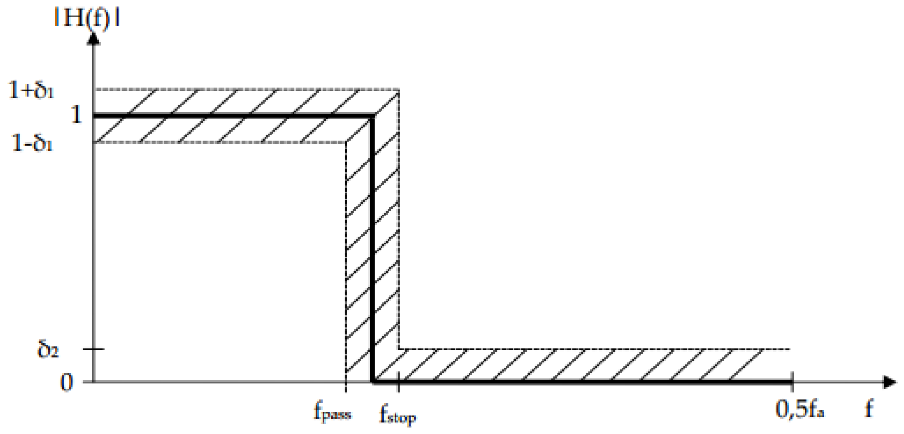

Figure 7: An ideal low-pass filter [3]

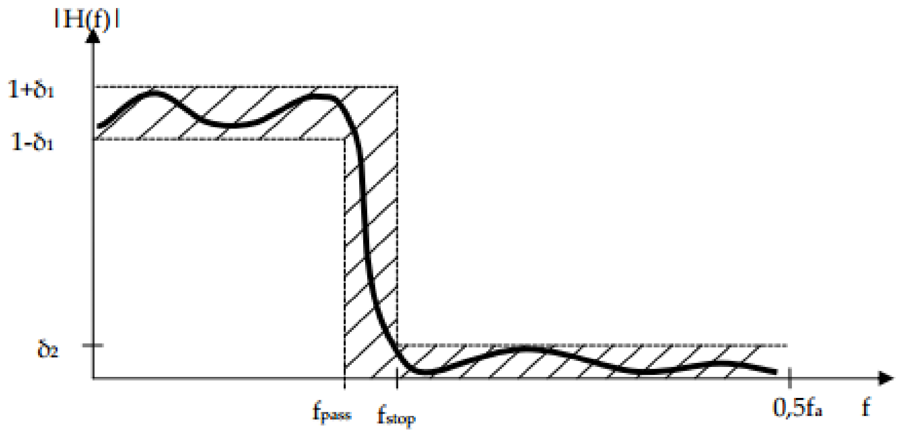

Figure 8: A low-pass filter in real life [3]

there are no information losses on the signal, if the sampling is done with the doubled maximum overtone frequency [3].

The height and therefore the frequency response $|H(f)|$ of the passed area is $1+/-\delta$, since we want the frequencies to pass as they are, with a tolerance of δ. However, in practice this behavior is not possible, as this is an ideal low-pass filter. It is not possible to implement such a filter, because of its square characteristics. A sinusoid which forms a perfect square requires an infinite order.

Figure 8 depicts a low-pass filter with a finite order which can occur in real-life systems. The higher the order the steeper the cut-off frequency.

One can see that the filter example in Section 2.1 is a low-pass, since it filters out the higher overtone frequency.

Besides the low-pass filter, there is also the high-pass filter with the opposite behavior. Ideally, a high-pass filter passes everything beyond the cut-off frequency $\frac{f_{stop}-f_{pass}}{2}$ and

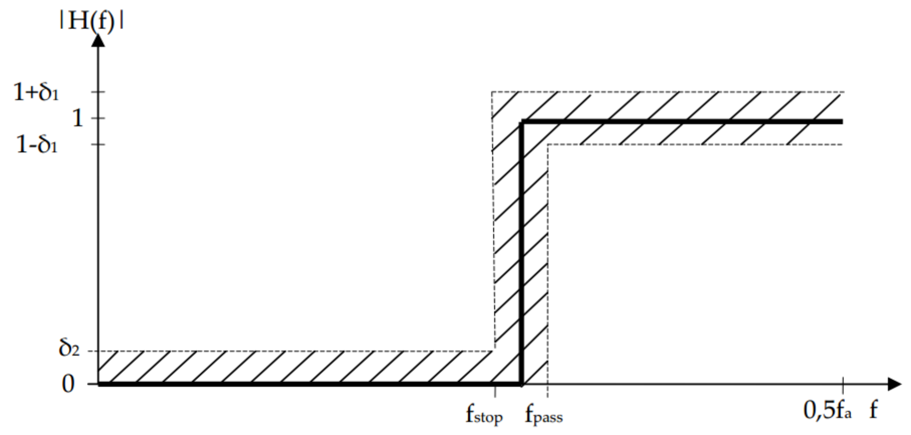

Figure 9: An ideal high-pass filter [3]

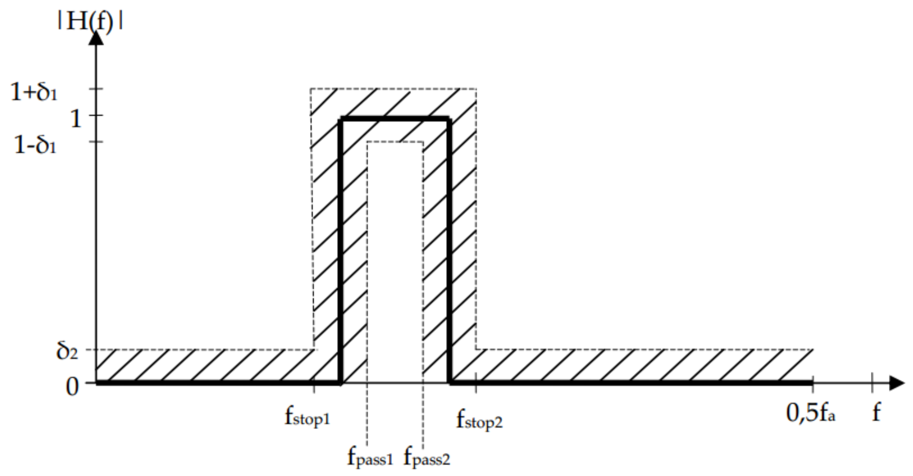

Figure 10: An ideal bandpass filter [3]

filters out everything below this threshold. This behavior is depicted in Figure 9.

Moreover there are 2 types of band filters which are widely used. These filters are not constraint to the leftmost or rightmost spectrum, i. e. frequency range. Every frequency beneath the lower cut-off frequency and beyond the upper cut-off frequency will be filtered. Only the range between f_{pass1} and f_{pass2} will be passed, as shown in Figure 10.

In practice, this filter is composed by designing and multiplying a low- and a high-pass filter.

In contrary to the bandpass filter, the bandstop filter passes each frequency beneath the lower cut-off frequency and beyond the upper cut-off frequency. Figure 11 depicts, that only a rather small spectrum will be filtered. These filters are often used to get rid of 50 Hz power supply noise. With the help of an ideal filter, this would be done by setting $f_{pass1} = 50 - \delta_1$ and $f_{pass1} = 50 + \delta_2$ with a small tolerance δ.

The band-stop filter can also be easily constructed by a low- and a high-pass filter. In order to achieve the band-stop functionality, these two filters need to be disjoint.

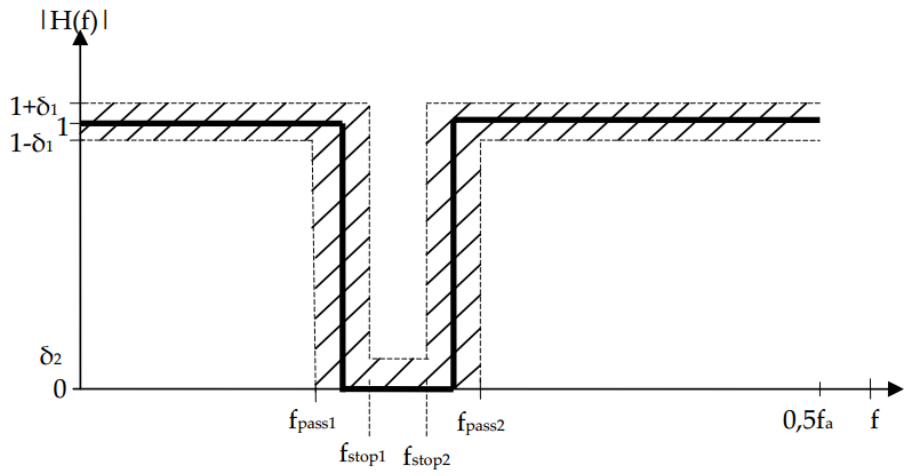

Figure 11: An ideal bandstop filter [3]

Th example from Section 2.1 can also be displayed in the frequency domain. Figure 12a illustrates the frequency range as seen in Section 2.2 (blue) and a low-pass filter with order 20 (red). The result of the multiplications of both signals is depicted in Figure 12b. After re-transforming it back to the time domain, we achieve the same output signal (Figure 12c), as we got in Section 2.1.

3 Digital Filters

So far, everything discussed applies on both analog and digital filters. As stated in [5], analog filters are cheap, fast and have a large dynamic range in amplitude and frequency. However, digital filters are vastly superior in the level of performance. Additionally, they also enable further realizations of the filter functions discussed in Section 2.3. These realizations are called Finite Impulse Response (FIR) and Infinite Impulse Response (IIR).

In General, the basis of these filters is the convolution $(h*n)(n)$. As discussed in Section 1, DSPs are defined by difference equations, which are the definition of the discrete convolution. This difference equation is illustrated in Equation 4 and works for discrete variable sequences.

$$(h*x)(n) = \sum_{m=-\infty}^{\infty} h(m)x(n-m) \tag{4}$$

3.1 Finite Impulse Response Filters

As pointed out above, FIR filters work with convolution. Furthermore, they have a finite impulse response. Hence the only impulse response amplitudes that may differ from 0 are in a specified, finite range. I. e. the impulse response of the FIR filter is limited. This is

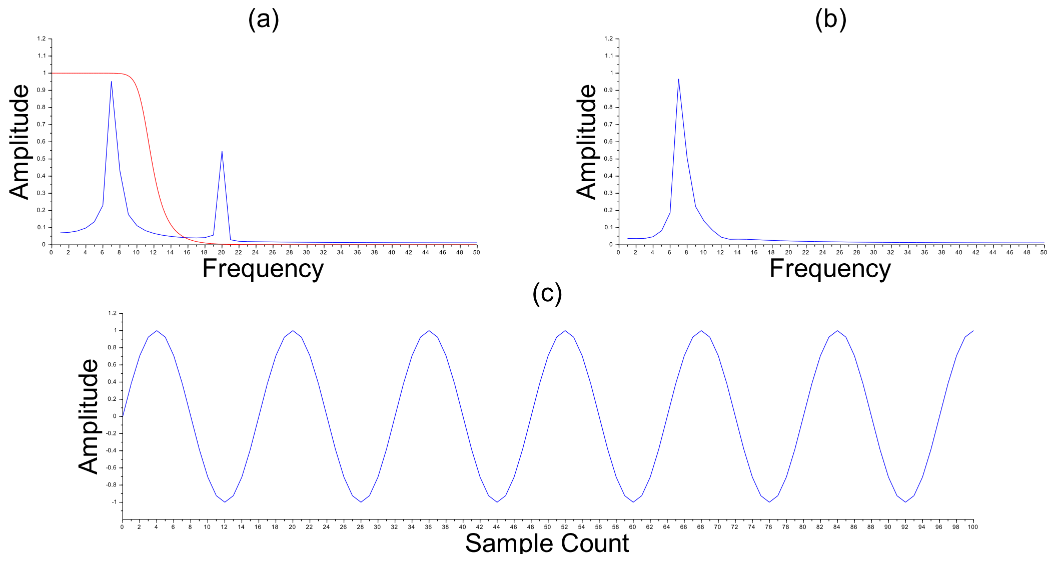

Figure 12: Low-pass filtering in frequency domain. (a) Input signal's (blue) and filter's (red) frequency domain (b) Filtered input signal in frequency domain (c) Re-transformed signal in time domain

the reason why the convolution can be given in a finite discrete difference equation, which can be seen in Equation 5.

$$y(n) = (x * h)(n) = \sum_{m=-\infty}^{\infty} h(m)x(n-m) = \sum_{m=0}^{M} b_m x(n-m) \tag{5}$$

The impulse response will be discretized and can be seen as the FIR's coefficients b_0 to b_M. The order of the FIR is then defined as $n = M - 1$ where M is first the number of coefficients and second the precision of the discretization. Figure 13 illustrates the behavior of a FIR filter, shown as a signal flow graph. The symbol z^{-1} means a delay of one sample, i. e. $z^{-1}x(n) = x(n-1)$ [4].

3.1.1 Filter Design

As stated in Section 2.2, the Discrete Fourier Transform of the impulse response yield the frequency response. And vice versa, the inverse Discrete Fourier Transform of the frequency response yield the impulse response. In order to describe the coefficients of the FIR filter, we need a proper impulse response, derived from the frequency response. Hence, one has to define the frequency response in the frequency domain, transfer it to the time domain and finally discretize the impulse response to get the coefficients.

However, the frequency response is often described as a periodic function. Since periodic functions are infinite, the re-transform using iDFT will also yield an infinite impulse response, which is in contrast to the definition of a FIR filter.

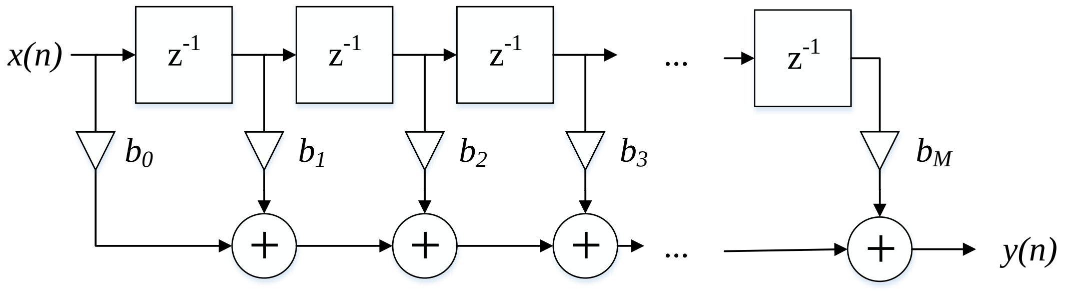

Figure 13: Signal flow graph of a FIR Filter with order $n = M - 1$, cf. [4]

As a result, when using periodic functions to describe the frequency response, one needs to apply window functions. The objective is to define a finite sample window to limit the impulse response. For instance, one can think of the rectangular window, where a signal with a group of adjacent points 0 to M has unity values, mostly 1, and 0 elsewhere [5]. Figure 14a depicts the rectangular window over a random sinusoid.

Another method is the hamming window, which is a smooth curve with $\frac{M}{2} = 1$. The hamming window is calculated from

$$hw(n) = 0.54 - 0.46 \cdot cos(\frac{2\pi n}{M}) \tag{6}$$

with $n = 0$ to M.

Figure 14b illustrates the Hamming window.

These windows are simply multiplied sample-wise to the filter to create the finite impulse response and furthermore the coefficients for the FIR filter.

3.2 Infinite Impulse Response Filters

Another type of digital filter is the analog inspired Infinite Response Filter (IIR) [2]. In comparison to a FIR, the IIR has an infinite impulse response. IIR filters can realize a subset of FIR filters, since the output depends on the input and the previous output. Hence, it uses feedback [2]. This is the reason why they are also called recursive filters. Besides the input signals $x(n)$, an IIR also depends on its own output $y(n-1)$. This is the reason for the infinite impulse response, which even may occur even though the input might have stopped [1].

Regarding the IIR difference equation in Equation 7, a FIR filter is special case of an IIR, with $a_i = 0$ for $i = 1..M$, where a_i are the feedback coefficients. Since the minuend is exactly the difference equation of a FIR (Equation 5). Thus, the subtrahend is the filter's feedback which is subtracted.

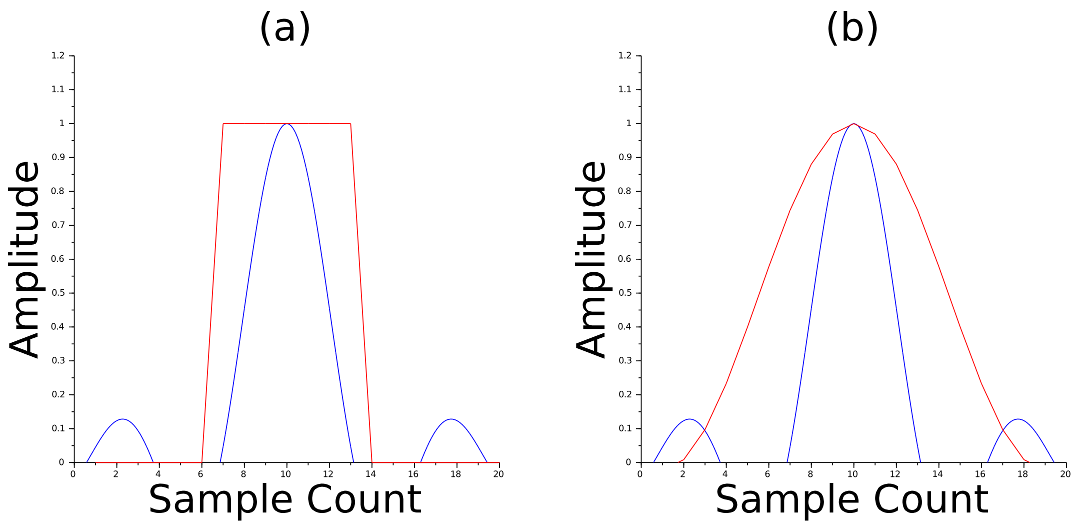

Figure 14: Illustration of window functions. (a) Rectangular window (b) Hamming window, cf. [4]

$$Y(n) = \sum_{m=0}^{M} b_m x(n-m) - \sum_{m=1}^{M} a_m x(n-m) \tag{7}$$

Figure 15 depicts the IIR's signal flow graph, based on its difference equation.

3.3 Comparison

As stated in [2] a FIR filter has several advantages over the IIR filter. First, they are easier to design and can be guaranteed to have a linear phase response, which is important for video and music processing. I. e. that the filter has constant group delay on the signals, whereas the phase response of the IIR is non-linear. Furthermore FIR filters have low sensitivity to filter coefficient quantization errors, which is important for implementing on an integrated circuit.

On the other hand, IIR filters are useful in a high-speed context, since they usually use less multiplications than FIR filters [2]. However, IIR filter do not have a linear phase response and can be unstable in case they are not designed properly, since post-pulse oscillation may occur.

4 Conclusion

This work presented the basics of different types of Digital Signal Processing, especially filtering techniques. It was pointed out, that filtering is possible in time and frequency domain. However, all of the techniques are based on convolution. Special attention was

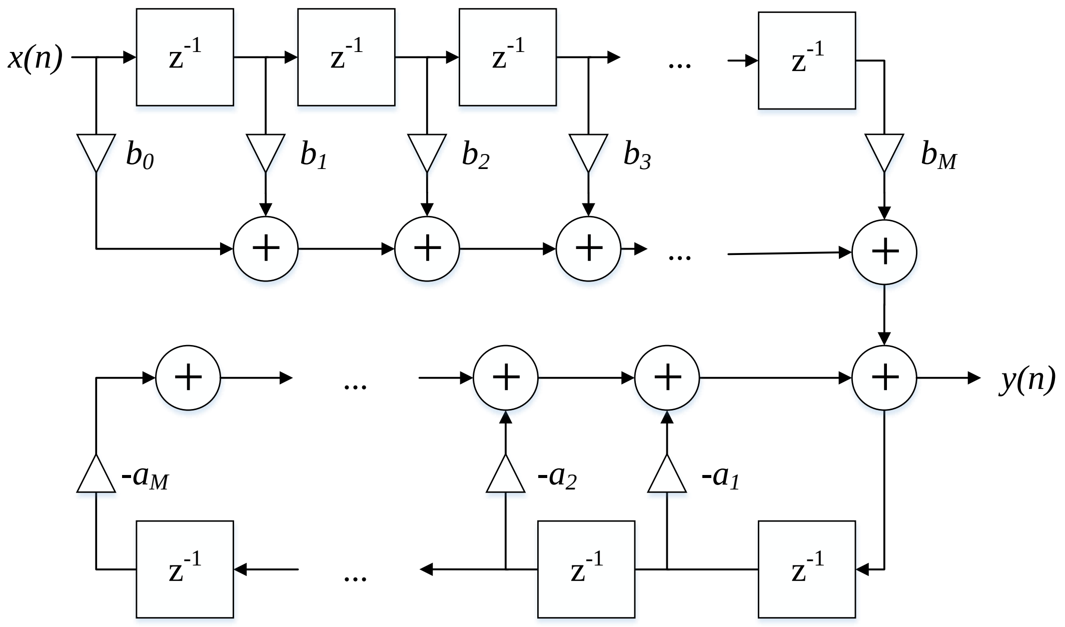

Figure 15: Signal flow graph of an IIR Filter with order $n = M - 1$, cf. [4]

paid to digital filters, like the Finite Impulse Response and the Infinite Impulse Response filter. Advantages of the FIR are the stable work with linear phase response, whereas the IIR is faster on integrated circuits.

Especially for audio systems on integrated circuits, one faces the trade-off between stability and speed. Altogether filtering is very important, most notably in audio applications, where effects are created in signals are adjusted.

References

[1] K. D. Kammeyer and K. Kroschel. *Digitale Signalverarbeitung*. Teubner Studienbücher, 4 edition, 1998.

[2] L. Litwin. FIR and IIR digital filters. *IEEE Potentials*, 19(4):28–31, 2000.

[3] H.W. Neuschwander. *Digitale Filter - Grundlagen und Programmierung*. 2010.

[4] Julius O. III Smith. *Introduction to Digital Filters - with Audio Applications*. Center for Computer Research in Music and Acoustics (CCRMA).

[5] SW Smith. *The scientist and engineer's guide to digital signal processing*. 1997.