2

Filters and delays

P. Dutilleux, M. Holters, S. Disch and U. Zölzer

2.1 Introduction

The term filter can have a large number of different meanings. In general it can be seen as a way to select certain elements with desired properties from a larger set. Let us focus on the particular field of digital audio effects and consider a signal in the frequency domain. The signal can be seen as a set of partials having different frequencies and amplitudes. The filter will perform a selection of the partials according to the frequencies that we want to reject, retain or emphasize. In other words: the filter will modify the amplitude of the partials according to their frequency. Once implemented, it will turn out that this filter is a linear transformation. As an extension, linear transformations can be said to be filters. According to this new definition of a filter, any linear operation could be said to be a filter, but this would go far beyond the scope of digital audio effects. It is possible to demonstrate what a filter is by using one's voice and vocal tract. Utter a vowel, *a*, for example, at a fixed pitch and then utter other vowels at the same pitch. By doing that we do not modify our vocal cords, but we modify the volume and the interconnection pattern of our vocal tract. The vocal cords produce a signal with a fixed harmonic spectrum, whereas the cavities act as acoustic filters to enhance some portions of the spectrum. We have described filters in the frequency domain here because it is the usual way to consider them, but they also have an effect in the time domain. After introducing a filter classification for basic filter types in the frequency domain, we will review typical implementation methods.

Beyond their effects in the frequency domain, filters can also be considered in the time domain leading to a family of delay-based audio effects such as those which can be experienced in acoustical spaces. A sound wave reflected by a wall will be superimposed on the sound wave at the source. If the wall is far away, such as a cliff, we will hear an echo. If the wall is close to us, we will notice the reflections through a modification of the sound color. Repeated reflections can appear between parallel boundaries. In a room, such reflections will be called *flatter echo*. The distance between the boundaries determines the delay that is imposed on each reflected sound wave. In a cylinder, successive reflections will develop at both ends. If the cylinder is long, we will hear

DAFX: Digital Audio Effects, Second Edition. Edited by Udo Zölzer.

an iterative pattern, whereas if the cylinder is short, we will hear a pitched tone. Equivalents of these acoustical phenomena have been implemented as signal processing units. In the case of a time-varying delay the direct physical correspondence can be a relative movement between sound source and listener imposing a frequency shift on the perceived signal, which is commonly referred to as the "Doppler effect."

2.2 Basic filters

2.2.1 Filter classification in the frequency domain

The various types of filters can be defined according to the following classification (see Figure 2.1):

- **Lowpass (LP)** filters select low frequencies up to the cut-off frequency f_c and attenuate frequencies higher than f_c. Additionally, a resonance may amplify frequencies around f_c.
- **Highpass (HP)** filters select frequencies higher than f_c and attenuate frequencies below f_c, possibly with a resonance around f_c.
- **Bandpass (BP)** filters select frequencies between a lower cut-off frequency f_{cl} and a higher cut-off frequency f_{ch}. Frequencies below f_{cl} and frequencies higher than f_{ch} are attenuated.
- **Bandreject (BR)** filters attenuate frequencies between a lower cut-off frequency f_{cl} and a higher cut-off frequency f_{ch}. Frequencies below f_{cl} and frequencies higher than f_{ch} are passed.
- **Allpass** filters pass all frequencies, but modify the phase of the input signal.

The lowpass with resonance is very often used in computer music to simulate an acoustical resonating structure; the highpass filter can remove undesired very low frequencies; the bandpass can produce effects such as the imitation of a telephone line or of a mute on an acoustical instrument; the bandreject can divide the audible spectrum into two bands that seem to be uncorrelated.

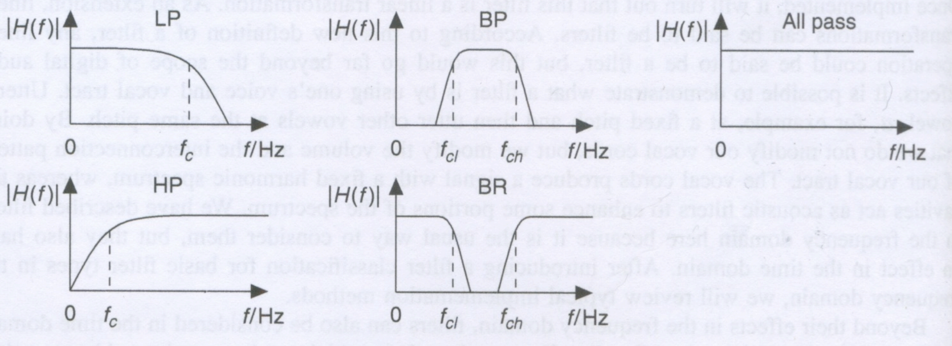

Figure 2.1 Filter classification.

2.2.2 Canonical filters

There are various ways to implement a filter, the simplest being the canonical filter, as shown in Figure 2.2 for a second-order filter, which can be implemented by the difference equations

$$x_h(n) = x(n) - a_1 x_h(n-1) - a_2 x_h(n-2) \tag{2.1}$$

$$y(n) = b_0 x_h(n) + b_1 x_h(n-1) + b_2 x_h(n-2) \tag{2.2}$$

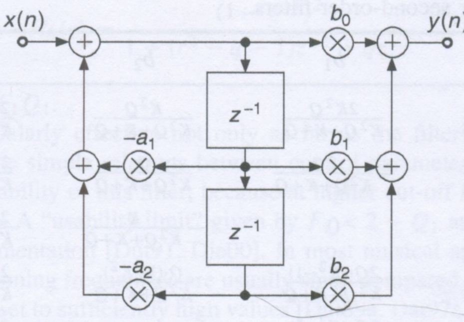

Figure 2.2 Canonical second-order digital filter.

Table 2.1 Coefficients for first-order filters.

	b_0	b_1	a_1
Lowpass	$K/(K+1)$	$K/(K+1)$	$(K-1)/(K+1)$
Highpass	$1/(K+1)$	$-1/(K+1)$	$(K-1)/(K+1)$
Allpass	$(K-1)/(K+1)$	1	$(K-1)/(K+1)$

and leads to the transfer function

$$H(z) = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2}}{1 + a_1 z^{-1} + a_2 z^{-2}}. \tag{2.3}$$

By setting $a_2 = b_2 = 0$, this reduces to a first-order filter which, can be used to implement an allpass, lowpass or highpass with the coefficients of Table 2.1 where K depends on the cut-off frequency f_c by

$$K = \tan(\pi f_c / f_S). \tag{2.4}$$

For the allpass filter, the coefficient K likewise controls the frequency f_c when $-90°$ phase shift is reached.

For the second-order filters with coefficients shown in Table 2.2, in addition to the cut-off frequency (for lowpass and highpass) or the center frequency (for bandpass, bandreject and allpass) we additionally need the Q factor with slightly different meanings for the different filter types:

- For the lowpass and highpass filters, it controls the height of the resonance. For $Q = \frac{1}{\sqrt{2}}$, the filter is maximally flat up to the cut-off frequency; for lower Q, it has higher pass-band attenuation, while for higher Q, amplification around f_c occurs.
- For the bandpass and bandreject filters, it is related to the bandwidth f_b by $Q = \frac{f_c}{f_b}$, i.e., it is the inverse of the relative bandwidth $\frac{f_b}{f_c}$.
- For the allpass filter, it likewise controls the bandwidth, which here depends on the points where $\pm$ 90° phase shift relative to the $-180°$ phase shift at f_c are reached.

While the canonical filters are relatively simple, the calculation of their coefficients from parameters like cut-off frequency and bandwidth is not. In the following, we will therefore study filter structures that are slightly more complicated, but allow for easier parameterization.

Table 2.2 Coefficients for second-order filters.

	b_0	b_1	b_2	a_1	a_2
Lowpass	$\frac{K^2Q}{K^2Q+K+Q}$	$\frac{2K^2Q}{K^2Q+K+Q}$	$\frac{K^2Q}{K^2Q+K+Q}$	$\frac{2Q\cdot(K^2-1)}{K^2Q+K+Q}$	$\frac{K^2Q-K+Q}{K^2Q+K+Q}$
Highpass	$\frac{Q}{K^2Q+K+Q}$	$-\frac{2Q}{K^2Q+K+Q}$	$\frac{Q}{K^2Q+K+Q}$	$\frac{2Q\cdot(K^2-1)}{K^2Q+K+Q}$	$\frac{K^2Q-K+Q}{K^2Q+K+Q}$
Bandpass	$\frac{K}{K^2Q+K+Q}$	0	$-\frac{K}{K^2Q+K+Q}$	$\frac{2Q\cdot(K^2-1)}{K^2Q+K+Q}$	$\frac{K^2Q-K+Q}{K^2Q+K+Q}$
Bandreject	$\frac{Q\cdot(1+K^2)}{K^2Q+K+Q}$	$\frac{2Q\cdot(K^2-1)}{K^2Q+K+Q}$	$\frac{Q\cdot(1+K^2)}{K^2Q+K+Q}$	$\frac{2Q\cdot(K^2-1)}{K^2Q+K+Q}$	$\frac{K^2Q-K+Q}{K^2Q+K+Q}$
Allpass	$\frac{K^2Q-K+Q}{K^2Q+K+Q}$	$\frac{2Q\cdot(K^2-1)}{K^2Q+K+Q}$	1	$\frac{2Q\cdot(K^2-1)}{K^2Q+K+Q}$	$\frac{K^2Q-K+Q}{K^2Q+K+Q}$

2.2.3 State variable filter

A nice alternative to the canonical filter structure is the state variable filter shown in Figure 2.3 [Cha80], which combines second-order lowpass, bandpass and highpass for the same f_c and Q. Its difference equation is given by

$$y_l(n) = F_1 y_b(n) + y_l(n-1) \tag{2.5}$$

$$y_b(n) = F_1 y_h(n) + y_b(n-1) \tag{2.6}$$

$$y_h(n) = x(n) - y_l(n-1) - Q_1 y_b(n-1), \tag{2.7}$$

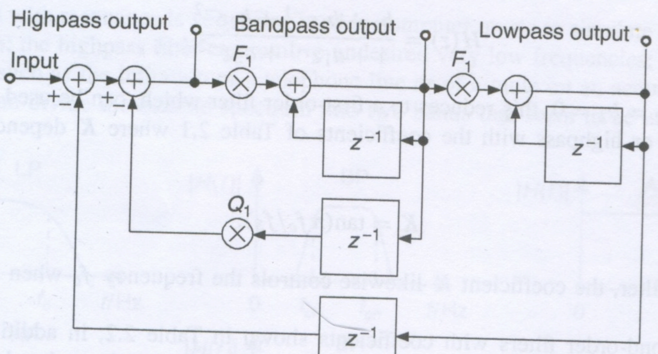

Figure 2.3 Digital state variable filter.

where $y_l(n)$, $y_b(n)$ and $y_h(n)$ are the outputs of lowpass, bandpass, and highpass, respectively. The tuning coefficients F_1 and Q_1 are related to the tuning parameters f_c and Q by

$$F_1 = 2\sin(\pi f_c/f_S) \qquad Q_1 = 1/Q. \tag{2.8}$$

It can be shown that the transfer function of lowpass, bandpass, and highpass, respectively, are

$$H_l(z) = \frac{r^2}{1+(r^2-q-1)z^{-1}+qz^{-2}} \tag{2.9}$$

$$H_b(z) = \frac{r\cdot(1-z^{-1})}{1+(r^2-q-1)z^{-1}+qz^{-2}} \tag{2.10}$$

$$H_h(z) = \frac{(1 - z^{-1})^2}{1 + (r^2 - q - 1)z^{-1} + qz^{-2}}, \tag{2.11}$$

with $r = F_1$ and $q = 1 - F_1 Q_1$.

This structure is particularly effective not only as far as the filtering process is concerned, but above all because of the simple relations between control parameters and tuning coefficients. One should consider the stability of this filter, because at higher cut-off frequencies and smaller Q factors it becomes unstable. A "usability limit" given by $F_1 < 2 - Q_1$ assures the stable operation of the state variable implementation [Dut91, Die00]. In most musical applications, however, it is not a problem because the tuning frequencies are usually small compared to the sampling frequency and the Q factor is usually set to sufficiently high values [Dut89a, Dat97a]. This filter has proven its suitability for a large number of applications. The nice properties of this filter have been exploited to produce endless glissandi out of natural sounds and to allow smooth transitions between extreme settings [Dut89b, m-Vas93]. It is also used for synthesizer applications [Die00].

2.2.4 Normalization

Filters are usually designed in the frequency domain and as a consequence, they are expected to primarily affect the frequency content of the signal. However, the side effect of loudness modification must not be forgotten because of its importance for the practical use of the filter. The filter might produce the right effect, but the result might be useless because the sound has become too weak or too strong. The method of compensating for these amplitude variations is called normalization. Normalization is performed by scaling the filter such that the norm

$$L_p = \left(\frac{1}{2\pi}\int_{-\pi}^{\pi} |H(e^{j\omega})|^p \mathrm{d}\omega\right)^{\frac{1}{p}} = 1, \tag{2.12}$$

where typically L_2 or $L_\infty = \max\left(|H(e^{j\omega})|\right)$ are used [Zöl05]. To normalize the loudness of the signal, the L_2 norm is employed. It is accurate for broadband signals and fits many practical musical applications. L_∞ normalizes the maximum of the frequency response and avoids overloading the filter. With a suitable normalization scheme the filter can prove to be very easy to handle whereas with the wrong normalization, the filter might be rejected by musicians because they cannot operate it.

The normalization of the state variable filter has been studied in [Dut91], where several effective implementation schemes are proposed. In practice, a first-order lowpass filter that processes the input signal will perform the normalization in f_c and an amplitude correction in $\sqrt{\zeta}$ will normalize in ζ (see Figure 2.4). This normalization scheme allows us to operate the filter with damping factors down to 10^{-4} where the filter gain reaches about 74 dB at f_c.

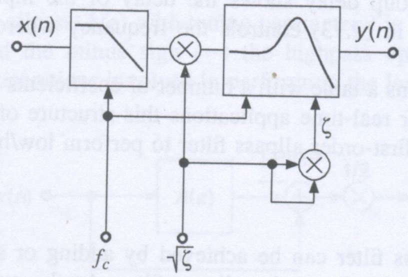

Figure 2.4 L_2-normalization in f_c and ζ for the state variable filter.

2.2.5 Allpass-based filters

In this subsection we introduce a special class of parametric filter structures for lowpass, highpass, bandpass and bandreject filter functions. Parametric filter structures denote special signal flow graphs where a coefficient inside the signal flow graph directly controls the cut-off frequency and bandwidth of the corresponding filter. These filter structures are easily tunable by changing only one or two coefficients. They play an important role for real-time control with minimum computational complexity.

The basis for parametric first- and second-order IIR filters is the first- and second-order allpass filter. We will first discuss the first-order allpass and show simple lowpass and highpass filters, which consist of a tunable allpass filter together with a direct path.

First-order allpass

A first-order allpass filter is given by the transfer function (see 2.2.2)

$$A(z) = \frac{z^{-1} + c}{1 + cz^{-1}} \tag{2.13}$$

$$c = \frac{\tan(\pi f_c/f_S) - 1}{\tan(\pi f_c/f_S) + 1} \tag{2.14}$$

and the corresponding difference equation

$$x_h(n) = x(n) - cx_h(n-1) \tag{2.15}$$

$$y(n) = cx_h(n) + x_h(n-1), \tag{2.16}$$

which can be realized by the block diagram shown in Figure 2.5.

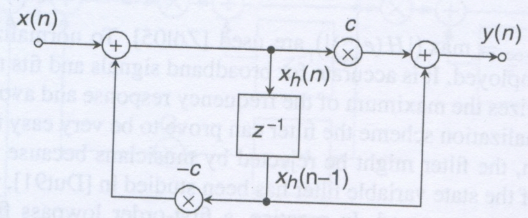

Figure 2.5 Block diagram for a first-order allpass filter.

The magnitude/phase response and the group delay of a first-order allpass are shown in Figure 2.6. The magnitude response is equal to one and the phase response is approaching −180° for high frequencies. The group delay shows the delay of the input signal in samples versus frequency. The coefficient c in (2.13) controls the frequency where the phase response passes −90° (see Figure 2.6).

For simple implementations a table with a number of coefficients for different cut-off frequencies is sufficient, but even for real-time applications this structure offers very few computations. In the following we use this first-order allpass filter to perform low/highpass filtering.

First-order low/highpass

A first-order lowpass/highpass filter can be achieved by adding or subtracting (+/−) the output signal from the input signal of a first-order allpass filter. As the output signal of the first-order allpass filter has a phase shift of −180° for high frequencies, this operation leads to low/highpass

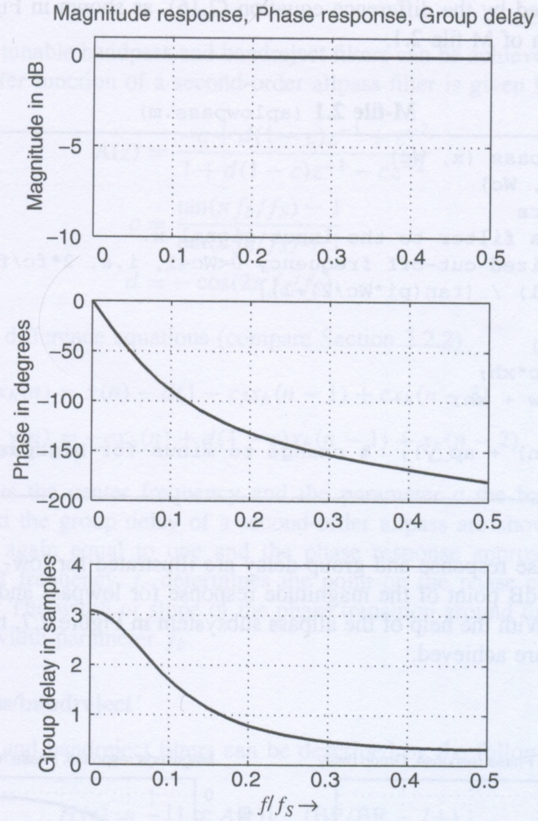

Figure 2.6 First-order allpass filter with $f_c = 0.1 \cdot f_S$.

filtering. The transfer function of a lowpass/highpass filter is then given by

$$H(z) = \frac{1}{2}\,(1 \pm A(z)) \quad (\text{LP/HP} + /-) \tag{2.17}$$

$$A(z) = \frac{z^{-1} + c}{1 + cz^{-1}} \tag{2.18}$$

$$c = \frac{\tan(\pi f_c/f_S) - 1}{\tan(\pi f_c/f_S) + 1}, \tag{2.19}$$

where a tunable first-order allpass $A(z)$ with tuning parameter c is used. The plus sign $(+)$ denotes the lowpass operation and the minus sign $(-)$ the highpass operation. The block diagram in Figure. 2.7 represents the operations involved in performing the low/highpass filtering. The allpass

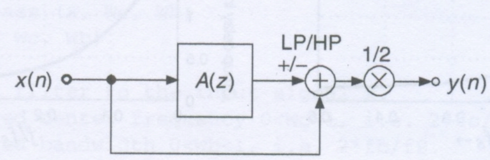

Figure 2.7 Block diagram of a first-order low/highpass filter.

filter can be implemented by the difference equation (2.16), as shown in Figure 2.5 to obtain the lowpass implementation of M-file 2.1.

M-file 2.1 (aplowpass.m)

```
function y = aplowpass (x, Wc)
% y = aplowpass (x, Wc)
% Author: M. Holters
% Applies a lowpass filter to the input signal x.
% Wc is the normalized cut-off frequency 0<Wc<1, i.e. 2*fc/fS.
c = (tan(pi*Wc/2)-1) / (tan(pi*Wc/2)+1);
xh = 0;
for n = 1:length(x)
  xh_new = x(n) - c*xh;
  ap_y = c * xh_new + xh;
  xh = xh_new;
  y(n) = 0.5 * (x(n) + ap_y);  % change to minus for highpass
end;
```

The magnitude/phase response and group delay are illustrated for low- and highpass filtering in Figure 2.8. The −3 dB point of the magnitude response for lowpass and highpass is passed at the cut-off frequency. With the help of the allpass subsystem in Figure 2.7, tunable first-order low- and highpass systems are achieved.

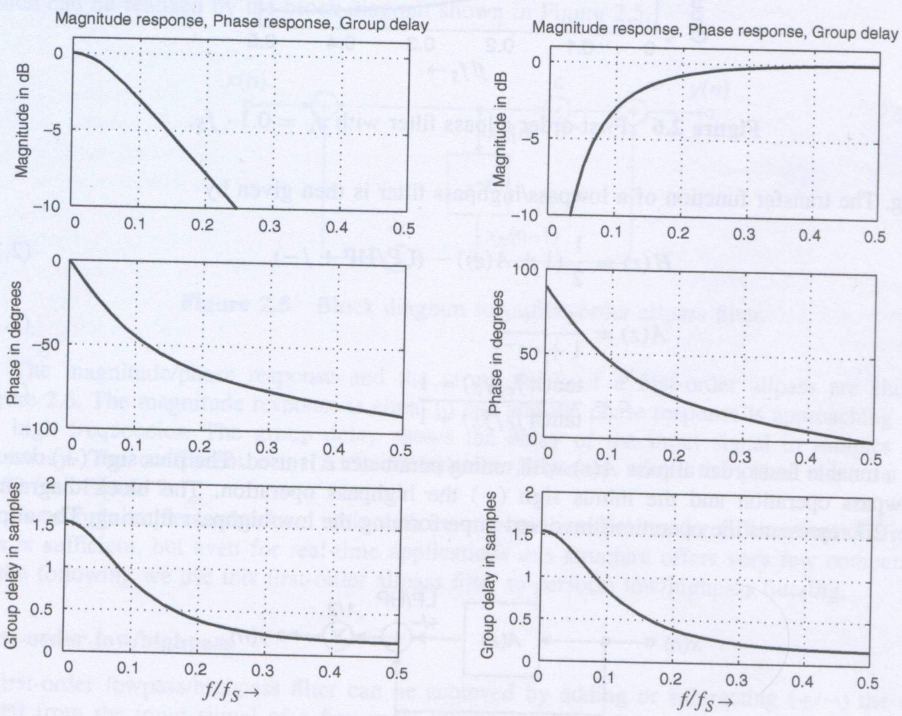

Figure 2.8 First-order low/highpass filter with $f_c = 0.1 f_S$.

Second-order allpass

The implementation of tunable bandpass and bandreject filters can be achieved with a second-order allpass filter. The transfer function of a second-order allpass filter is given by

$$A(z) = \frac{-c + d(1-c)z^{-1} + z^{-2}}{1 + d(1-c)z^{-1} - cz^{-2}} \tag{2.20}$$

$$c = \frac{\tan(\pi f_b/f_S) - 1}{\tan(\pi f_b/f_S) + 1} \tag{2.21}$$

$$d = -\cos(2\pi f_c/f_S), \tag{2.22}$$

with the corresponding difference equations (compare Section 2.2.2)

$$x_h(n) = x(n) - d(1-c)x_h(n-1) + cx_h(n-2) \tag{2.23}$$

$$y(n) = -cx_h(n) + d(1-c)x_h(n-1) + x_h(n-2). \tag{2.24}$$

The parameter d adjusts the center frequency and the parameter c the bandwidth. The magnitude/phase response and the group delay of a second-order allpass are shown in Figure 2.9. The magnitude response is again equal to one and the phase response approaches $-360°$ for high frequencies. The cut-off frequency f_c determines the point on the phase curve where the phase response passes $-180°$. The width or slope of the phase transition around the cut-off frequency is controlled by the bandwidth parameter f_b.

Second-order bandpass/bandreject

Second-order bandpass and bandreject filters can be described by the following transfer function

$$H(z) = \frac{1}{2}\,[1 \mp A(z)] \quad (\mathrm{BP/BR} - /+) \tag{2.25}$$

$$A(z) = \frac{-c + d(1-c)z^{-1} + z^{-2}}{1 + d(1-c)z^{-1} - cz^{-2}} \tag{2.26}$$

$$c = \frac{\tan(\pi f_b/f_S) - 1}{\tan(\pi f_b/f_S) + 1} \tag{2.27}$$

$$d = -\cos(2\pi f_c/f_S), \tag{2.28}$$

where a tunable second-order allpass $A(z)$ with tuning parameters c and d is used. The minus sign ($-$) denotes the bandpass operation and the plus sign ($+$) the bandreject operation. The block diagram in Figure 2.10 shows the bandpass and bandreject filter implementation based on a second-order allpass subsystem, M-file 2.2 shows the corresponding MATLAB® code. The magnitude/phase response and group delay are illustrated in Figure 2.11 for both filter types.

M-file 2.2 (apbandpass.m)

```
function y = apbandpass (x, Wc, Wb)
% y = apbandpass (x, Wc, Wb)
% Author: M. Holters
% Applies a bandpass filter to the input signal x.
% Wc is the normalized center frequency 0<Wc<1, i.e. 2*fc/fS.
% Wb is the normalized bandwidth 0<Wb<1, i.e. 2*fb/fS.
c = (tan(pi*Wb/2)-1) / (tan(pi*Wb/2)+1);
d = -cos(pi*Wc);
```

```
xh = [0, 0];
for n = 1:length(x)
  xh_new = x(n) - d*(1-c)*xh(1) + c*xh(2);
  ap_y = -c * xh_new + d*(1-c)*xh(1) + xh(2);
  xh = [xh_new, xh(1)];
  y(n) = 0.5 * (x(n) - ap_y);  % change to plus for bandreject
end;
```

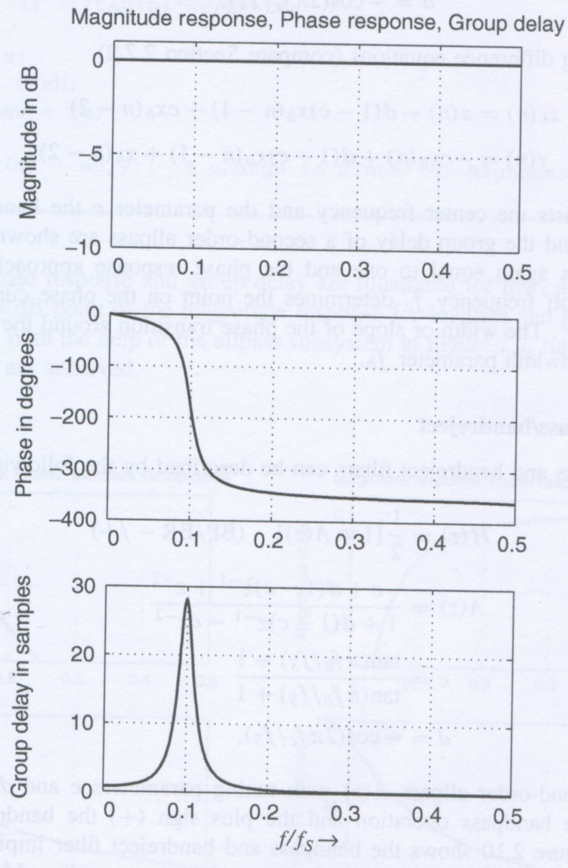

Figure 2.9 Second-order allpass filter with $f_c = 0.1 f_S$ and $f_b = 0.022 f_S$.

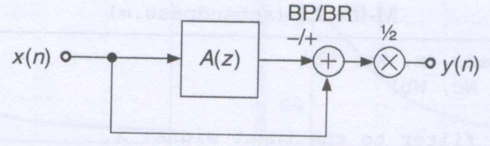

Figure 2.10 Second-order bandpass and bandreject filter.

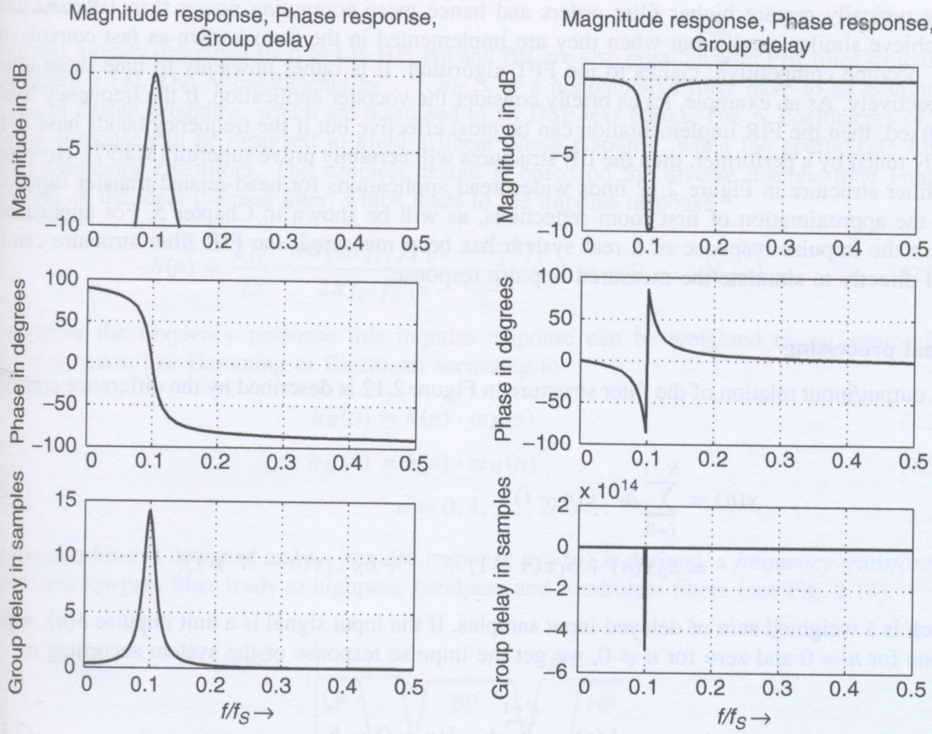

Figure 2.11 Second-order bandpass/bandreject filter with $f_c = 0.1 f_S$ and $f_b = 0.022 f_S$.

2.2.6 FIR filters

Introduction

The digital filters that we have seen before are said to have an infinite impulse response. Because of the feedback loops within the structure, an input sample will excite an output signal whose duration is dependent on the tuning parameters and can extend over a fairly long period of time. There are other filter structures without feedback loops (Figure 2.12). These are called finite impulse response filters (FIR), because the response of the filter to a unit impulse lasts only for a fixed period of time. These filters allow the building of sophisticated filter types where strong attenuation of unwanted frequencies or decomposition of the signal into several frequency bands is necessary.

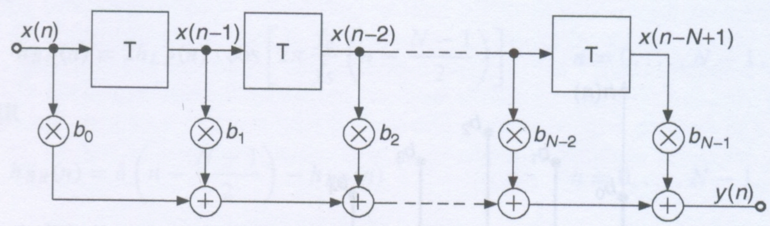

Figure 2.12 Finite impulse response filter.

They typically require higher filter orders and hence more computing power than IIR structures to achieve similar results, but when they are implemented in the form known as fast convolution they become competitive, thanks to the FFT algorithm. It is rather unwieldy to tune these filters interactively. As an example, let us briefly consider the vocoder application. If the frequency bands are fixed, then the FIR implementation can be most effective but if the frequency bands have to be subtly tuned by a performer, then the IIR structures will certainly prove superior [Mai97]. However, the filter structure in Figure 2.12 finds widespread applications for head-related transfer functions and the approximation of first room reflections, as will be shown in Chapter 5. For applications where the impulse response of a real system has been measured, the FIR filter structure can be used directly to simulate the measured impulse response.

Signal processing

The output/input relation of the filter structure in Figure 2.12 is described by the difference equation

$$y(n) = \sum_{i=0}^{N-1} b_i \cdot x(n-i) \tag{2.29}$$

$$= b_0 x(n) + b_1 x(n-1) + \cdots + b_{N-1} x(n-N+1), \tag{2.30}$$

which is a weighted sum of delayed input samples. If the input signal is a unit impulse $\delta(n)$, which is one for $n = 0$ and zero for $n \neq 0$, we get the impulse response of the system according to

$$h(n) = \sum_{i=0}^{N-1} b_i \cdot \delta(n-i) = b_n. \tag{2.31}$$

A graphical illustration of the impulse response of a five-tap FIR filter is shown in Figure 2.13. The Z-transform of the impulse response gives the transfer function

$$H(z) = \sum_{i=0}^{N-1} b_i \cdot z^{-i} \tag{2.32}$$

and with $z = \mathrm{e}^{j\omega}$ the frequency response

$$H(e^{j\omega}) = b_0 + b_1 \mathrm{e}^{-j\omega} + b_2 \mathrm{e}^{-j2\omega} + \cdots + b_{N-1} \cdot \mathrm{e}^{-j(N-1)\omega} \tag{2.33}$$

$$\text{with} \quad \omega = 2\pi f / f_S.$$

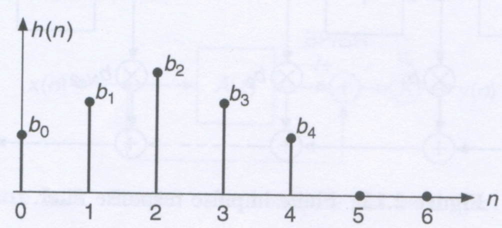

Figure 2.13 Impulse response of an FIR filter.

Filter design

The filters already described such as LP, HP, BP and BR are also possible with FIR filter structures (see Figure 2.14). The N coefficients $b_0, \ldots, b_{N-1}$ of a nonrecursive filter have to be computed by special design programs, which are discussed in all DSP text books. The N coefficients of the impulse response can be designed to yield a linear phase response, when the coefficients fulfill certain symmetry conditions. The simplest design is based on the inverse discrete-time Fourier transform of the ideal lowpass filter, which leads to the impulse response

$$h(n) = \frac{2f_c}{f_S} \cdot \frac{\sin\left[2\pi f_c/f_S\left(n - \frac{N-1}{2}\right)\right]}{2\pi f_c/f_S\left(n - \frac{N-1}{2}\right)}, \quad n = 0, \ldots, N-1. \tag{2.34}$$

To improve the frequency response this impulse response can be weighted by an appropriate window function like Hamming or Blackman according to

$$h_B(n) = h(n) \cdot w_B(n) \tag{2.35}$$

$$h_H(n) = h(n) \cdot w_H(n) \tag{2.36}$$

$$n = 0, 1, \ldots, N-1.$$

If a lowpass filter is designed and an impulse response $h_{LP}(n)$ is derived, a *frequency transformation* of this lowpass filter leads to highpass, bandpass and bandreject filters (see Fig. 2.14).

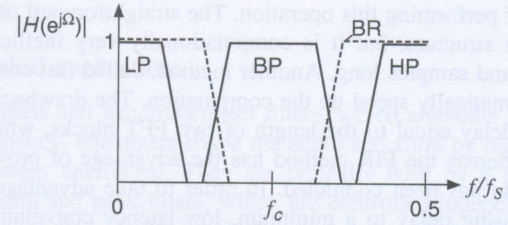

Figure 2.14 Frequency transformations: LP and frequency transformations to BP and HP.

Frequency transformations are performed in the time domain by taking the lowpass impulse response $h_{LP}(n)$ and computing the following equations:

- LP-HP

$$h_{HP}(n) = h_{LP}(n) \cdot \cos\left[\pi\left(n - \frac{N-1}{2}\right)\right] \qquad n = 0, \ldots, N-1 \tag{2.37}$$

- LP-BP

$$h_{BP}(n) = 2h_{LP}(n) \cdot \cos\left[2\pi \frac{f_c}{f_S}\left(n - \frac{N-1}{2}\right)\right] \qquad n = 0, \ldots, N-1 \tag{2.38}$$

- LP-BR

$$h_{BR}(n) = \delta\left(n - \frac{N-1}{2}\right) - h_{BP}(n) \qquad n = 0, \ldots, N-1. \tag{2.39}$$

Another simple FIR filter design is based on the FFT algorithm and is called *frequency sampling*. Its main idea is to specify the desired frequency response at discrete frequencies uniformly distributed over the frequency axis and calculate the corresponding impulse response. Design examples for audio processing with this design technique can be found in [Zöl05].

Musical applications

If linear phase processing is required, FIR filter offer magnitude equalization without phase distortions. They allow real-time equalization by making use of the frequency sampling design procedure [Zöl05] and are attractive equalizer counterparts to IIR filters, as shown in [McG93]. A discussion of more advanced FIR filters for audio processing can be found in [Zöl05].

2.2.7 Convolution

Introduction

Convolution is a generic signal processing operation like addition or multiplication. In the realm of computer music, however, it has the particular meaning of imposing a spectral or temporal structure onto a sound. These structures are usually not defined by a set of a few parameters, such as the shape or the time response of a filter, but are given by a signal which lasts typically a few seconds or more. Although convolution has been known and used for a very long time in the signal-processing community, its significance for computer music and audio processing has grown with the availability of fast computers that allow long convolutions to be performed in a reasonable period of time.

Signal processing

We could say in general that the convolution of two signals means filtering one with the other. There are several ways of performing this operation. The straightforward method is a direct implementation in a FIR filter structure, but it is computationally very inefficient when the impulse response is several thousand samples long. Another method, called fast convolution, makes use of the FFT algorithm to dramatically speed up the computation. The drawback of fast convolution is that it has a processing delay equal to the length of two FFT blocks, which is objectionable for real-time applications, whereas the FIR method has the advantage of providing a result immediately after the first sample has been computed. In order to take advantage of the FFT algorithm while keeping the processing delay to a minimum, low-latency convolution schemes have been developed which are suitable for real-time applications [Gar95, MT99].

The result of convolution can be interpreted in both the frequency and time domains. If $a(n)$ and $b(n)$ are the two convolved signals, the output spectrum will be given by the product of the two spectra $S(f) = A(f) \cdot B(f)$. The time interpretation derives from the fact that if $b(n)$ is a pulse at time k, we will obtain a copy of $a(n)$ shifted at time k_0, i.e., $s(n) = a(n-k)$. If $b(n)$ is a sequence of pulses, we will obtain a copy of $a(n)$ in correspondence to every pulse, i.e., a rhythmic, pitched or reverberated structure, depending on the pulse distance. If $b(n)$ is pulse-like, we obtain the same pattern with a filtering effect. In this case $b(n)$ should be interpreted as an impulse response. Thus convolution will result in subtractive synthesis, where the frequency shape of the filter is determined by a real sound. For example convolution with a bell sound will be heard as filtered by the resonances of the bell. In fact the bell sound is generated by a strike on the bell and can be considered as the impulse response of the bell. In this way we can simulate the effect of a sound hitting a bell, without measuring the resonances and designing the filter. If both sounds $a(n)$ and $b(n)$ are complex in time and frequency, the resulting sound will be blurred and will tend to lack the original sound's character. If both sounds are of long duration and each has a strong pitch and smooth attack, the result will contain both pitches and the intersection of their spectra.

Musical applications

When a sound is convolved with the impulse responses of a room, it is projected in the corresponding virtual auditory space [DMT99]. A diffuse reverberation can be produced by convolving with broad-band noise having a sharp attack and exponentially decreasing amplitude. Another example is obtained by convolving a tuba glissando with a series of snare-drum strokes. The tuba is transformed into something like a Tibetan trumpet playing in the mountains. Each stroke of the snare drum produces a copy of the tuba sound. Since each stroke is noisy and broadband, it acts like a reverberator. The series of strokes acts like several diffusing boundaries and produces the type of echo that can be found in natural landscapes [DMT99].

The convolution can be used to map a rhythm pattern onto a sampled sound. The rhythm pattern can be defined by positioning a unit impulse at each desired time within a signal block. The convolution of the input sound with the time pattern will produce copies of the input signal at each of the unit impulses. If the unit impulse is replaced by a more complex sound, each copy will be modified in its timbre and in its time structure. If a snare drum stroke is used, the attacks will be smeared and some diffusion will be added. The convolution has an effect both in the frequency and in the time domain. Take a speech sound with sharp frequency resonances and a rhythm pattern defined by a series of snare-drum strokes. Each word will appear with the rhythm pattern and the rhythm pattern will be heard in each word with the frequency resonances of the initial speech sound.

Convolution as a tool for musical composition has been investigated by composers such as Horacio Vaggione [m-Vag96, Vag98] and Curtis Roads [Roa97]. Because convolution has a combined effect in the time and frequency domains, some expertise is necessary to foresee the result of the combination of two sounds.

2.3 Equalizers

Introduction and musical applications

In contrast to low/highpass and bandpass/reject filters, which attenuate the audio spectrum above or below a cut-off frequency, equalizers shape the audio spectrum by enhancing certain frequency bands while others remain unaffected. They are typically built by a series connection of first- and second-order shelving and peak filters, which are controlled independently (see Figure 2.15). Shelving filters boost or cut the low- or high-frequency bands with the parameters cut-off frequency f_c and gain G. Peak filters boost or cut mid-frequency bands with parameters center frequency f_c, bandwidth f_b and gain G. One often-used filter type is the constant Q peak filter. The Q factor is defined by the ratio of the bandwidth to center frequency $Q = \frac{f_b}{f_c}$. The center frequency of peak filters is then tuned while keeping the Q factor constant. This means that the bandwidth is increased when the center frequency is increased and vice versa. Several proposed digital filter structures for shelving and peak filters can be found in the literature [Whi86, RM87, Dut89a, HB93, Bri94, Orf96, Orf97, Zöl05].

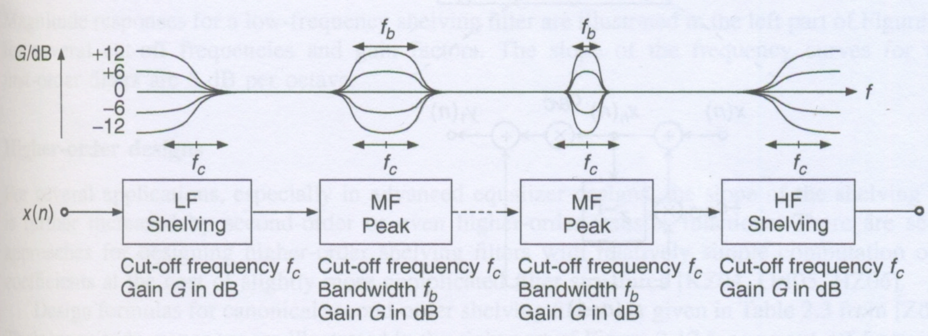

Figure 2.15 Series connection of shelving and peak filters.

Applications of these parametric filters can be found in parametric equalizers, octave equalizers ($f_c = 31.25, 62.5, 125, 250, 500, 1000, 2000, 4000, 8000, 16\,000$ Hz) and all kinds of equalization devices in mixing consoles, outboard equipment and foot-pedal controlled stomp boxes.

2.3.1 Shelving filters

First-order design

Similar to the first-order lowpass/highpass filters described in Section 2.2.5, first-order low/high frequency shelving filters can be constructed based on a first-order allpass [Zöl05], yielding the transfer function

$$H(z) = 1 + \frac{H_0}{2}\,[1 + \pm A(z)] \quad \text{(LF/HF} + / -) \tag{2.40}$$

with the first-order allpass

$$A(z) = \frac{z^{-1} + c_{B/C}}{1 + c_{B/C}z^{-1}}. \tag{2.41}$$

The block diagram in Figure 2.16 shows a first-order low/high-frequency shelving filter, which leads to the difference equations

$$x_h(n) = x(n) - c_{B/C}x_h(n-1) \tag{2.42}$$

$$y_1(n) = c_{B/C}x_h(n) + x_h(n-1) \tag{2.43}$$

$$y(n) = \frac{H_0}{2}\left[x(n) \pm y_1(n)\right] + x(n). \tag{2.44}$$

The gain G in dB for low/high frequencies can be adjusted by the parameter

$$H_0 = V_0 - 1 \quad \text{with} \quad V_0 = 10^{G/20}. \tag{2.45}$$

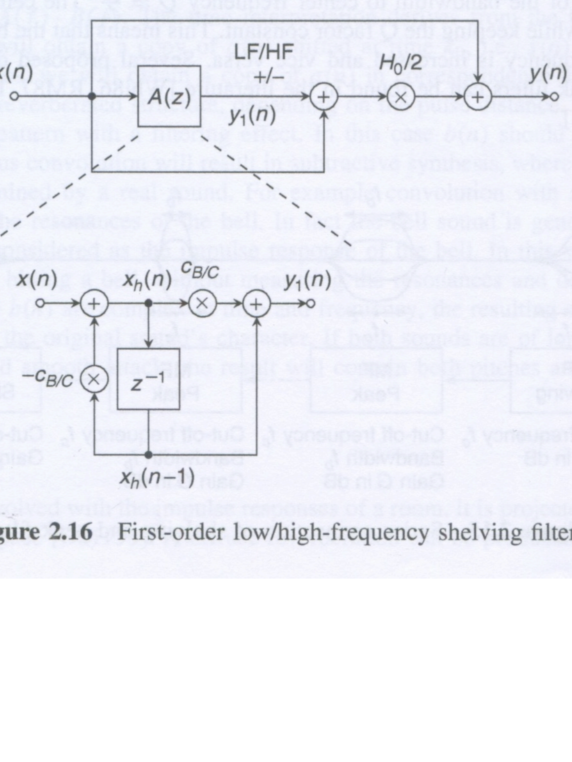

Figure 2.16 First-order low/high-frequency shelving filter.

The cut-off frequency parameters, c_B for boost and c_C for cut, for a first-order low-frequency shelving filter can be calculated [Zöl05] as

$$c_B = \frac{\tan(\pi f_c/f_S) - 1}{\tan(\pi f_c/f_S) + 1}, \tag{2.46}$$

$$c_C = \frac{\tan(\pi f_c/f_S) - V_0}{\tan(\pi f_c/f_S) + V_0} \tag{2.47}$$

and for a high-frequency shelving filter as

$$c_B = \frac{\tan(\pi f_c/f_S) - 1}{\tan(\pi f_c/f_S) + 1}$$

$$c_C = \frac{V_0 \tan(\pi f_c/f_S) - 1}{V_0 \tan(\pi f_c/f_S) + 1}.$$

An implementation of this approach is given in M-file 2.3

M-file 2.3 (lowshelving.m)

```
function y = lowshelving (x, Wc, G)
% y = lowshelving (x, Wc, G)
% Author: M. Holters
% Applies a low-frequency shelving filter to the input signal x.
% Wc is the normalized cut-off frequency 0<Wc<1, i.e. 2*fc/fS
% G is the gain in dB
V0 = 10^(G/20); H0 = V0 - 1;
if G >= 0
  c = (tan(pi*Wc/2)-1) / (tan(pi*Wc/2)+1);     % boost
else
  c = (tan(pi*Wc/2)-V0) / (tan(pi*Wc/2)+V0);   % cut
end;
xh = 0;
for n = 1:length(x)
  xh_new = x(n) - c*xh;
  ap_y = c * xh_new + xh;
  xh = xh_new;
  y(n) = 0.5 * H0 * (x(n) + ap_y) + x(n);  % change to minus for HS
end;
```

Magnitude responses for a low-frequency shelving filter are illustrated in the left part of Figure 2.17 for several cut-off frequencies and gain factors. The slope of the frequency curves for these first-order filters are 6 dB per octave.

Higher-order designs

For several applications, especially in advanced equalizer designs, the slope of the shelving filter is further increased by second-order or even higher-order transfer functions. There are several approaches for designing higher-order shelving filters with relatively simple computation of the coefficients at the cost of slightly more complicated filter structures [KZ04, Orf05, HZ06].

Design formulas for canonical second-order shelving filters are given in Table 2.3 from [Zöl05]. Their magnitude responses are illustrated in the right part of Figure 2.17 for two cut-off frequencies and several gain factors.

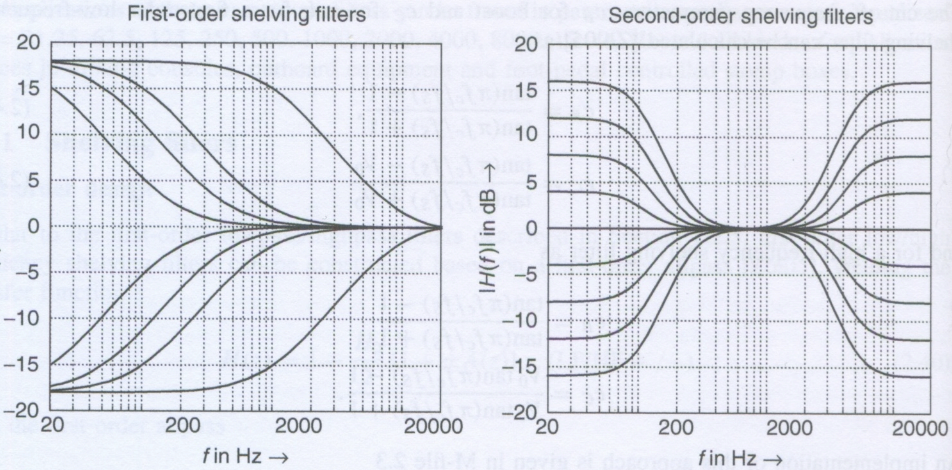

Figure 2.17 Frequency responses for first-order and second-order shelving filters.

Table 2.3 Second-order shelving filter design with $K = \tan(\pi f_c/f_S)$ and $V_0 = 10^{G/20}$ [Zöl05].

	b_0	b_1	b_2	a_1	a_2
LF boost	$\frac{1+\sqrt{2V_0}K+V_0K^2}{1+\sqrt{2}K+K^2}$	$\frac{2(V_0K^2-1)}{1+\sqrt{2}K+K^2}$	$\frac{1-\sqrt{2V_0}K+V_0K^2}{1+\sqrt{2}K+K^2}$	$\frac{2(K^2-1)}{1+\sqrt{2}K+K^2}$	$\frac{1-\sqrt{2}K+K^2}{1+\sqrt{2}K+K^2}$
LF cut	$\frac{V_0(1+\sqrt{2}K+K^2)}{V_0+\sqrt{2V_0}K+K^2}$	$\frac{2V_0(K^2-1)}{V_0+\sqrt{2V_0}K+K^2}$	$\frac{V_0(1-\sqrt{2}K+K^2)}{V_0+\sqrt{2V_0}K+K^2}$	$\frac{2(K^2-V_0)}{V_0+\sqrt{2V_0}K+K^2}$	$\frac{V_0-\sqrt{2V_0}K+K^2}{V_0+\sqrt{2V_0}K+K^2}$
HF boost	$\frac{V_0+\sqrt{2V_0}K+K^2}{1+\sqrt{2}K+K^2}$	$\frac{2(K^2-V_0)}{1+\sqrt{2}K+K^2}$	$\frac{V_0-\sqrt{2V_0}K+K^2}{1+\sqrt{2}K+K^2}$	$\frac{2(K^2-1)}{1+\sqrt{2}K+K^2}$	$\frac{1-\sqrt{2}K+K^2}{1+\sqrt{2}K+K^2}$
HF cut	$\frac{V_0(1+\sqrt{2}K+K^2)}{1+\sqrt{2V_0}K+V_0K^2}$	$\frac{2V_0(K^2-1)}{1+\sqrt{2V_0}K+V_0K^2}$	$\frac{V_0(1-\sqrt{2}K+K^2)}{1+\sqrt{2V_0}K+V_0K^2}$	$\frac{2(V_0K^2-1)}{1+\sqrt{2V_0}K+V_0K^2}$	$\frac{1-\sqrt{2V_0}K+V_0K^2}{1+\sqrt{2V_0}K+V_0K^2}$

2.3.2 Peak filters

Similarly, a second-order peak filter [Zöl05] is given by the transfer function

$$H(z) = 1 + \frac{H_0}{2}\left[1 - A_2(z)\right], \tag{2.48}$$

where

$$A_2(z) = \frac{-c_{B/C} + d(1 - c_{B/C})z^{-1} + z^{-2}}{1 + d(1 - c_{B/C})z^{-1} - c_{B/C}z^{-2}} \tag{2.49}$$

is a second-order allpass filter. The block diagram in Figure 2.18 shows the second-order peak filter, which leads to the difference equations

$$x_h(n) = x(n) - d(1 - c_{B/C})x_h(n-1) + c_{B/C}x_h(n-2) \tag{2.50}$$

$$y_1(n) = -c_{B/C}x_h(n) + d(1 - c_{B/C})x_h(n-1) + x_h(n-2) \tag{2.51}$$

$$y(n) = \frac{H_0}{2}\left[x(n) - y_1(n)\right] + x(n). \tag{2.52}$$

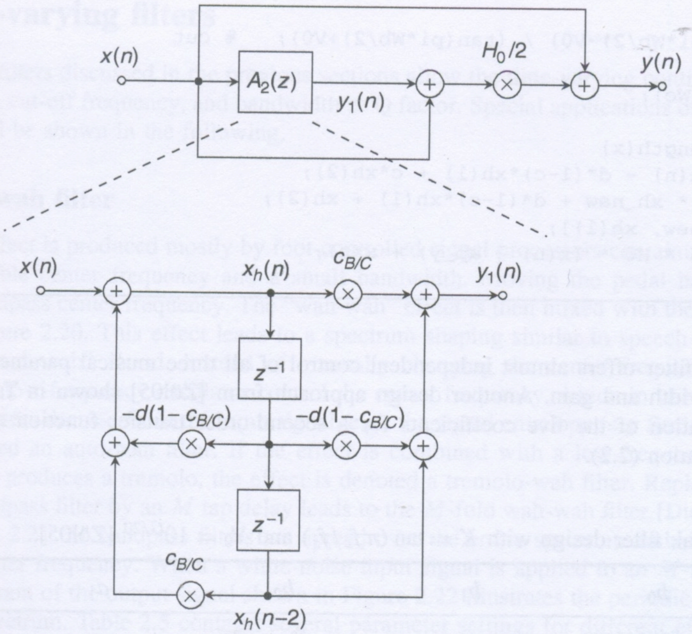

Figure 2.18 Second-order peak filter.

The center frequency parameter d and the coefficient H_0 are given by

$$d = -\cos(2\pi f_c/f_S) \tag{2.53}$$

$$V_0 = H(e^{j2\pi f_c/f_S}) = 10^{G/20} \tag{2.54}$$

$$H_0 = V_0 - 1. \tag{2.55}$$

The bandwidth f_b is adjusted through the parameters c_B and c_C for boost and cut given by

$$c_B = \frac{\tan(\pi f_b/f_S) - 1}{\tan(\pi f_b/f_S) + 1} \tag{2.56}$$

$$c_C = \frac{\tan(\pi f_b/f_S) - V_0}{\tan(\pi f_b/f_S) + V_0}. \tag{2.57}$$

A possible peak filter implementation using this approach is given in M-file 2.4.

M-file 2.4 (peakfilt.m)

```
function y = peakfilt (x, Wc, Wb, G)
% y = peakfilt (x, Wc, Wb, G)
% Author: M. Holters
% Applies a peak filter to the input signal x.
% Wc is the normalized center frequency 0<Wc<1, i.e. 2*fc/fS.
% Wb is the normalized bandwidth 0<Wb<1, i.e. 2*fb/fS.
% G is the gain in dB.
V0 = 10^(G/20); H0 = V0 - 1;
if G >= 0
  c = (tan(pi*Wb/2)-1) / (tan(pi*Wb/2)+1);     % boost
```

```
else
  c = (tan(pi*Wb/2)-V0) / (tan(pi*Wb/2)+V0);   % cut
end;
d = -cos(pi*Wc);
xh = [0, 0];
for n = 1:length(x)
  xh_new = x(n) - d*(1-c)*xh(1) + c*xh(2);
  ap_y = -c * xh_new + d*(1-c)*xh(1) + xh(2);
  xh = [xh_new, xh(1)];
  y(n) = 0.5 * H0 * (x(n) - ap_y) + x(n);
end;
```

This peak filter offers almost independent control of all three musical parameters center frequency, bandwidth and gain. Another design approach from [Zöl05] shown in Table 2.4 allows direct computation of the five coefficients for a second-order transfer function as given in the difference equation (2.2).

Table 2.4 Peak filter design with $K = \tan(\pi f_c/f_s)$ and $V_0 = 10^{G/20}$ [Zöl05].

	b_0	b_1	b_2	a_1	a_2
Boost	$\frac{1+\frac{V_0}{Q}K+K^2}{1+\frac{1}{Q}K+K^2}$	$\frac{2(K^2-1)}{1+\frac{1}{Q}K+K^2}$	$\frac{1-\frac{V_0}{Q}K+K^2}{1+\frac{1}{Q}K+K^2}$	$\frac{2(K^2-1)}{1+\frac{1}{Q}K+K^2}$	$\frac{1-\frac{1}{Q}K+K^2}{1+\frac{1}{Q}K+K^2}$
Cut	$\frac{1+\frac{1}{Q}K+K^2}{1+\frac{1}{V_0Q}K+K^2}$	$\frac{2(K^2-1)}{1+\frac{1}{V_0Q}K+K^2}$	$\frac{1-\frac{1}{Q}K+K^2}{1+\frac{1}{V_0Q}K+K^2}$	$\frac{2(K^2-1)}{1+\frac{1}{V_0Q}K+K^2}$	$\frac{1-\frac{1}{V_0Q}K+K^2}{1+\frac{1}{V_0Q}K+K^2}$

Frequency responses for several settings of a peak filter are shown in Figure 2.19. The left part shows a variation of the gain with a fixed center frequency and bandwidth. The right part shows for fixed gain and center frequency a variation of the bandwidth or Q factor.

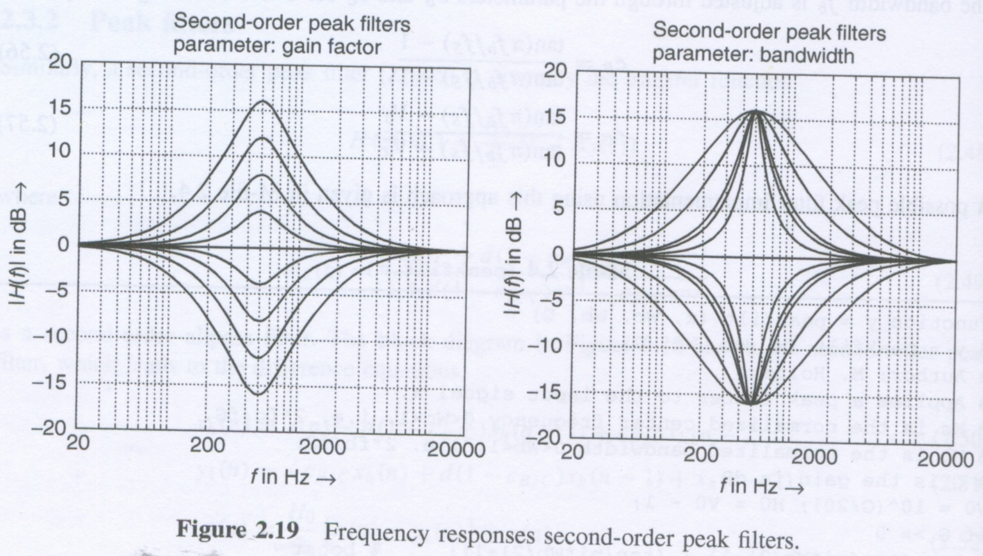

Figure 2.19 Frequency responses second-order peak filters.