

Robert Anderson
DST - Fall 2005
Final Report

Basic Low-Pass Filter Design

It may seem illogical for someone who has had only one or two basic electronics classes and little or no math background to study filter design, but I am fascinated by the ongoing analog versus digital debate as well as the debate as to whether or not higher sampling rates should really make a difference in the quality of audio. While it is generally accepted that audio recorded at higher sampling rates sounds better (all other things being equal), the debate seems centered on how this could be. Even humans with exceptional hearing cannot perceive sound above 23 kHz, and the accepted sampling theorem states that any signal can be accurately reconstructed provided that it is sampled at twice the highest signal frequency, and that frequencies above this desired limit are filtered out before sampling.

Some proponents of high sampling rates put forth the theory that ultrasonic frequencies can be perceived, whether or not humans can actually "hear" them, that the influence of these frequencies on the signal affect the audible sound regardless of the fact that these components of the sound are well above what any person, even one with exceptional hearing, can hear. Many of those who hold this position claim that analog recordings (tape or vinyl) are the true measure of sound, and that higher sampling rates are simply closer to the (theoretically) infinite frequency resolution of analog. While this theory may have some merit, it seems to me that it cannot be the only reason. The high frequency response of many of the best studio microphones is limited to, or falls short of, 20 kHz; much of the analog recording and playback equipment also has limited high frequency response. It would seem that there is another reason for this perceived increase of quality at higher sampling rates.

It is the feeling of many equipment designers that the affect of low-pass filtering on the sound may be to blame. The author's own theories of the affect of bias frequencies on analog tape and inverse RIAA curves on phonograph playback equipment aside, there is much evidence that the need for a "brickwall" filter response to eliminate frequencies above the Nyquist limit may have a profound effect on the quality of the sampled audio. This has led me on a quest to better understand the design of low-pass filters - analog and digital - so that I can have a better understanding of the technology at play and therefore make intelligent and well-informed decisions as the technology continues to evolve. Since I have only a very basic understanding of electronics, it seemed a good idea to study the most basic filters first, and then work my way up to more complicated designs.

Audio filters can be broken into two basic groups: acoustic and electronic.

Electronic filters can also be broken down into two groups: analog and digital. Each of these categories can again be broken down into two groups: analog filters can be active or passive; digital filters can be divided into finite and infinite impulse response filters. While the implementation differs greatly from category to category, the core design principals are common to each type of filter, and a filter with a specific response can be implemented in almost any of the four electronic categories that are of interest here.

Passive Filters

Passive filters are made of the three basic building blocks of electronic circuits: resistors, capacitors, and inductors. Passive filters do not rely on any external power source - they simply filter the signal as it passes through the circuit. The behaviour of a passive circuit is completely dependent on impedance - resistance and reactance. Therefore the key to understanding electronic filters is having an understanding of impedance. Impedance is complex - it is made up of resistance and reactance. Resistance is constant at all frequencies whereas reactance depends on frequency.

The most elementary low-pass filter is an inductor - essentially just a wire coil. An inductor will not block DC (which can be considered 0 Hz) and its reactance will increase as AC frequency increases:

$$\text{Inductive Reactance} = X_L = 2\pi fL$$

f = frequency; L = inductance (measured in Henries)

A capacitor will have the opposite response. DC will not pass through a capacitor, but its reactance will *decrease* as AC frequency increases

$$\text{Capacitive Reactance} = X_C = -(1/2\pi fC)$$

C = capacitance (measured in farads).

Simply put: capacitive reactance is the inverse of inductive reactance. It can be seen that as f increases, inductive reactance will increase and capacitive reactance will decrease. This reactance is measured in Ohms and is combined with resistance to determine the response of a circuit at a particular frequency. Impedance is represented by a vector with a complex value made up of resistance, which provides the real value, and reactance which provides the imaginary value. To calculate total impedance, we simply need to find the length of the vector Z :

$$Z^2 = R^2 + X^2$$

To determine the frequency response of a filter, we use a transfer function, which takes the basic form:

$$\text{Gain}(f) = V_{\text{out}} / V_{\text{in}}$$

To be able to calculate V_{out} at frequency f , we need to calculate the length of the impedance vector at frequency f . The length and angle of this vector will change depending on the particular combination of reactance and resistance at a particular frequency.

Tuning the Circuit

The complex nature of impedance leads to some interesting phenomena. Of interest in this case: a resistor could be used in conjunction with an inductor (RL filter) or a capacitor (RC filter) to tune a circuit to a particular frequency. At the frequency where the reactance is equal to the resistance, the response of the circuit will change. This frequency is called the cutoff frequency or the corner frequency.

In the case of a low-pass filter, we could place an inductor in series with the load or we could place a capacitor in parallel. With an RL filter, when $f = 0$, the inductor acts as a short circuit allowing all of the current to pass through. In this case $V_{\text{out}} = V_{\text{in}}$. At very high frequencies the inductor acts as an open circuit: all of the voltage drops across the inductor rather than passing through to the resistor. With an inductor in series, lower frequencies pass through to the load while high frequencies are choked off.

On the other hand, if we place a capacitor in parallel with the load and the current will follow the path of least resistance so to speak. At low frequencies, the capacitor acts as an open circuit: all of the current will flow through the load, while at very high frequencies, the reactance will be small compared to the resistor. Most of the current would flow through the capacitor, bypassing the load.

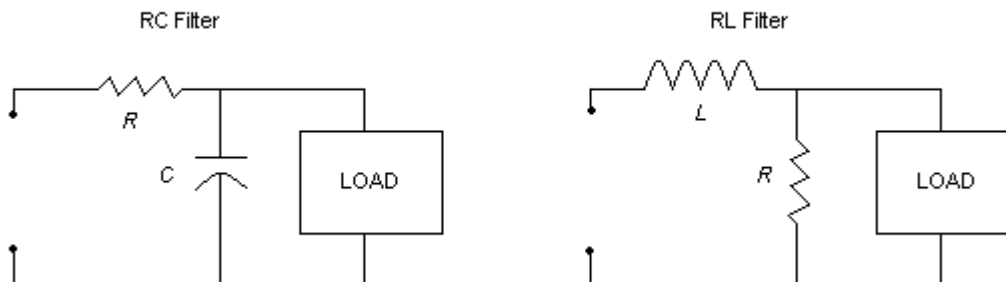


Fig. 1 - Basic low-pass filter schematics

The corner or cutoff frequency is calculated according to the formulae:

$$\text{RC filter} = f_c = 1/2\pi RC$$

$$\text{RL filter} = f_c = R/2\pi L$$

Cutoff frequency is not really an accurate name for this frequency. This would only be true in the case of an ideal "brickwall filter." Instead, at this corner frequency, the voltage will be about 70.7% of its maximum (peak) value. The key is that all frequencies above this "cutoff point" will be attenuated whereas frequencies below this point will pass through more or less unimpeded.

If we have an RC low-pass filter, or an RL high-pass filter, we can determine the relative gain or loss at a particular frequency in decibels by using the equation:

$$N_{dBV} = 20 \log(X_C/Zt)$$

With an RL low-pass or an RC high-pass filter, we change the equation a little:

$$N_{dBV} = 20 \log(R/Zt)$$

In either case, if we were to plot the slope of our results, we would find that for a low-pass filter with a single capacitor or inductor, there would be two asymptotes intersecting at the corner frequency - one would be flat from 0 Hz to the corner frequency, and the other would have a slope of -6 dB. In the transition between these two intersecting lines there would be a curve with a dip of -1dB at one octave *below* the corner frequency, a dip of -3dB at the corner frequency, and the frequency one octave higher than the corner frequency would actually be at -7dB. This diagram is called a Bode plot.

If we wished to achieve a more dramatic attenuation of frequencies above the cutoff point, we could use some of these reactive components together. This would raise the order of the filter: two reactive components would make a second order filter (-12dB/octave), three make a third order filter (-18 dB/octave), four would form a fourth order filter (-24 dB/octave), and so on. Generally the rolloff for an n order filter is $(6 * n)$ dB per octave.

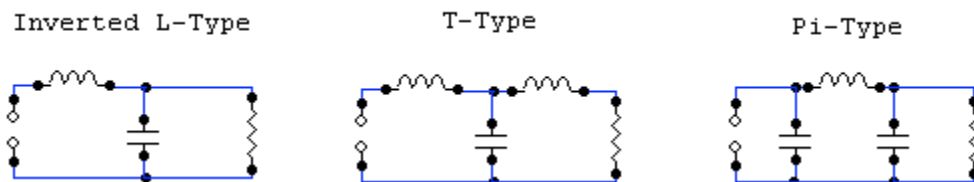


Fig. 2 - Different higher-order passive filter types [1]

The inverted-L type in figure 2 is a second-order filter, the T-type and the pi-type are third-order filters. Using a combination of inductors and capacitors opens up another Pandora's box: circuit resonance. A detailed look would be beyond the scope of this paper but where $X_L = X_C$, the circuit will be come resonant at that frequency. "At resonance, a parallel-tuned cicuit behaves like an open circuit...a series-tuned circuit behaves like a short." [22] (p.21-1)

Active filters

Audio electronics often use active filters as opposed to passive filters. The reason for this has as much to do with cost as it is to do with the complex nature of impedance. Cost is easy to understand: inductors for audio frequencies are big, heavy, and expensive, whereas capacitors, resistors and op amps are small, light, and relatively cheap. The impedance issue is not immediately intuitive.

The load on the other side of the filter can interact with the impedance of the filter. Also, cascading two filters one after the other would give us a sharper cutoff slope, but the impedance of these filters can also interact. The best way to eliminate the interaction between the stages would be to somehow isolate one from the other. An active filter uses electronic components that have the ability to amplify, such as transistors or op amps, to accomplish this.

Op amps have two inputs: an inverting input and a non-inverting input. The inverting input changes the polarity of the input by 180 degrees. The output voltage is determined by the difference between the two input voltages (differential input voltage = V_{diff}) and the gain factor of the op amp. The difference voltage is amplified by the voltage gain of the op amp to give us the output voltage.

An op amp has a huge multiplication factor, meaning that the output voltage is potentially huge. In reality, this output is limited by the voltage that is supplied to the op amp - known as the voltage rails. This limit is referred to as the *saturation voltage*. The output of the op amp has to lie in between these two rails. If the output voltage lies within the limits of the saturation voltage, then the differential input voltage must be so small that it can be considered zero for all practical purposes.

In order to keep the circuit stable, and to prevent complete saturation of the op amp, negative feedback is used. To achieve negative feedback, a path is opened between the output and the inverting input. The negative feedback allows the op amp to keep a lid on its own output. In other words, negative feedback effectively reduces the overall gain of the op amp circuit.

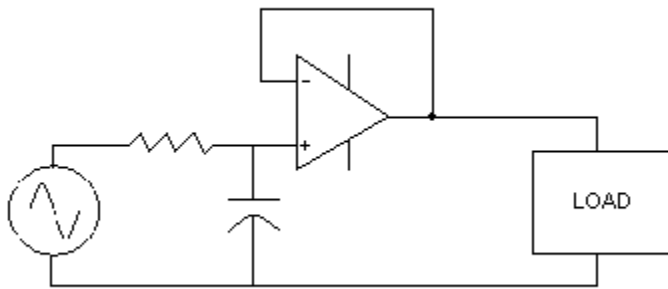


Fig. 3 - Active low-pass filter

In Figure 3, we have a simple RC filter connected to the non-inverting input of an op amp. The op amp has no resistors in the feedback path, meaning that it has the maximum amount of feedback possible sent to the inverting input. There will be no difference between the inputs, so the circuit is set up for a gain of 1. We are not trying to amplify the signal - the op amp simply isolates the filter from the load.

The input signal does not have to supply the current for the load. Also, the op amp circuit has an output impedance that is nearly zero. The load will see a perfectly clean voltage source with no output impedance, eliminating interaction between the impedance at the two stages. Grob states: "The main reason for using the op amp is that the RC filter can be isolated from the load, which may have a very low impedance...because Z_{in} is so high looking into the op amp and because Z_{out} is approximately zero, the RC filter is effectively isolated from the load." (p.946) As far as getting higher-order filters using an op amp: "one can get a second order filter without concern about one section loading another section." [12] (p.19) Of course, using the op amp also gives the potential for frequency-selective amplification, but that is beyond the scope of this paper.

Poles, Zeros, and the s-plane

Poles and zeros are at the very heart of both analog and digital filter design. Poles and zeros are relatively easy to define, but not so easy to conceptualize. Peter Stark [11] gives a good explanation: "the expression ' $s / (s-2)$ ' becomes infinite if s equals 2 because the denominator then becomes 0...so there is a pole at $s = 2$. On the other hand, the expression is zero if $s = 0$, so there is a zero at $s = 0$." (p.20-4)

A pole is a very powerful thing mathematically, since infinity is a rather large number. In the filtering world it would take a great number of zeros to accomplish what a just a couple of poles can do.

Both axis in the s-plane are measured in Hertz, the horizontal axis is labelled σ , and the vertical axis is labelled $j\omega$. By placing poles and zeros on or around these axes, we determine the response of our filter. Usually, a pole is placed on

the negative axis of σ . An oscillator occurs at frequency $j\omega$ if σ is equal to zero. If σ is less than 0, we have a stable system - one whose value would decrease with time, while if σ is greater than zero, our system would be unstable - the ringing would increase over time like uncontrolled feedback in a PA system.

In the case of a simple low-pass filter, we could place a single pole (X in Figure 4) on the horizontal axis on the negative side, whose absolute value is equal to that of the corner frequency. We could then take any other frequency and plot it on the vertical axis ("B"), and then measure the distance from one point to the other. To find the gain or loss at that particular frequency, we would divide the distance of the pole from the vertical axis ("A") by the distance from point A to point B ("D")

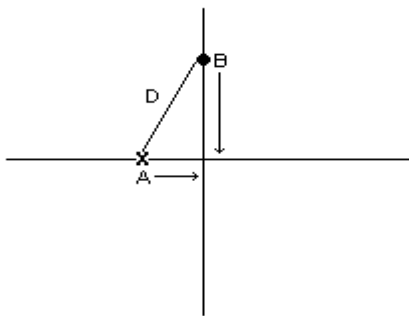


Fig. 4 - the s-plane [11]

According to [11] "Each capacitor or inductor in a filter circuit introduces a pole" (p.20-4) This single pole would then represent a first-order filter. Since we are dealing with a first-order filter, we would expect a drop of 7dB one octave above the corner frequency. We can set our corner frequency to 10kHz on the horizontal axis and our one-octave-higher frequency would end up at 20kHz on the vertical axis.

$$\begin{aligned} D &= \sqrt{A^2 + B^2} \\ &= \sqrt{(10\text{kHz})^2 + (20\text{kHz})^2} \\ &= 22361 \end{aligned}$$

$$\begin{aligned} \text{Gain} &= A / D \\ &= 10\text{kHz} / 22,361 \\ &= 0.447 \end{aligned}$$

$$\begin{aligned} \text{Gain}_{\text{dBV}} &= 20 \log(A / D) \\ &= 20 \log(0.447) \\ &= - 6.994 \text{ dB} \end{aligned}$$

Since each pole represents a capacitor or inductor in the filter, each pole in the s-plane represents yet another filter order. Therefore if we had two poles, we would have a second-order filter. For this reason, a second-order filter is

sometimes referred to as a two-pole filter. We could put the two poles on the same point on the σ axis, but by moving them off of the real axis, we can further manipulate the response of the filter.

Manipulating the position of the poles in the s-plane involves manipulating something called Q, which stands for quality factor. The Q is dependent on the amount of resistance in a circuit - lower resistance = higher Q. "Decreasing the Q moves the poles toward each other, whereas increasing the Q moves the poles in a semicircle away from each other and towards the $j\omega$ axis." [23] (p.6) This comes into play especially in the case of bandpass or notch filters - which is a topic for another paper. I was able to implement my filters without having a firm understanding of Q, though I will continue to study this in the future.

We can actually think of the two poles as two separate filters [11]. As in Figure 4, we would calculate the distance from point B to each individual pole, giving us D1 and D2. The resulting gain is simply a product of the two different gains. The tradeoff is: the placement of the poles will affect the frequency response in the passband. To illustrate this, let's take a look at some common low-pass filters.

Butterworth and Chebyshev Filters

When Butterworth and Chebyshev filters are implemented as low-pass filters, they are "all pole filters, meaning that their transfer functions contain all poles. The polynomial, which characterizes the filter's response, is used in the denominator of the filter's transfer functions. The polynomial's zeros are thus the filter's poles." [13] (p.4)

Butterworth filters are defined in [13] as: "maximally-flat-magnitude-response filters, optimized for gain flatness in the passband." (p.3) and are prized in the audio world for this flat passband response. This is accomplished by rotating the poles on a circle about the origin whose radius is equal to the length of the cutoff frequency. In the case of Figure 5, we have a 2-pole or second order Butterworth filter. The two poles have been rotated ± 45 degrees:

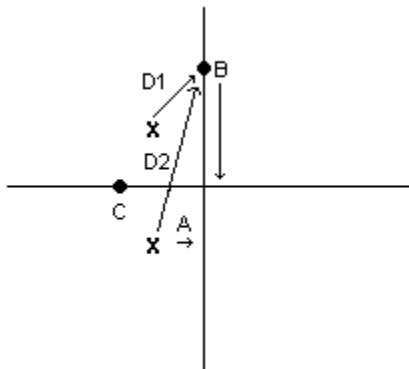


Fig. 5 - pole locations for a second-order Butterworth filter [11]

The distance from the pole to the origin is still our cutoff frequency 10 kHz ("C") , but we can see that "A" has changed. "A" is still the distance from the pole to the vertical axis which in this case becomes 7070 Hz (0.707 * 10 kHz). The distance "B" (still 20kHz here) will be plus or minus this amount when we do our calculations. To find the gain of this more complex filter we first need to find D_1 and D_2 :

$$D_1 = \sqrt{(20 \text{ kHz} - 7070\text{Hz})^2 + 7070^2} = 14737$$

$$D_2 = \sqrt{(20 \text{ kHz} + 7070)^2 + 7070^2} = 27978$$

But we use the cutoff frequency "C" when we figure the gain of each filter:

$$\begin{aligned} \text{Gain} &= \text{Gain}_1 * \text{Gain}_2 \\ &= (C / D_1) * (C / D_2) \\ &= C^2 / D_1 * D_2 \\ &= 10 \text{ kHz}^2 / 14737 * 27978 \\ &= 0.242 \end{aligned}$$

$$\text{Gain}_{\text{DBV}} = 20 \log(0.242) = -12.3 \text{ dB}$$

By using two poles instead of one, we have an attenuation of -12dB per octave in the stopband, which is consistent with the expected behaviour of a second-order filter. However, I expected to have at least -13dB at one octave above the cutoff since normally the frequency one octave above the corner frequency would be 1dB below the asymptote. This would imply to me that perhaps the Butterworth filter does not have as sharp a cutoff slope as a typical second-order filter.

In order to design a Butterworth filter for our needs, we would need to know certain criteria: the transfer function of a low-pass Butterworth filter, desired passband gain for a given range of frequencies, and desired stopband attenuation above a given frequency. Using a method described in [15] let us design a Butterworth filter that approximates the filter we have above: a pass band gain of -1dB for $0 < \omega < 500 \text{ Hz}$ and a stop band attenuation of no more than -13 dB at about 2000 Hz.

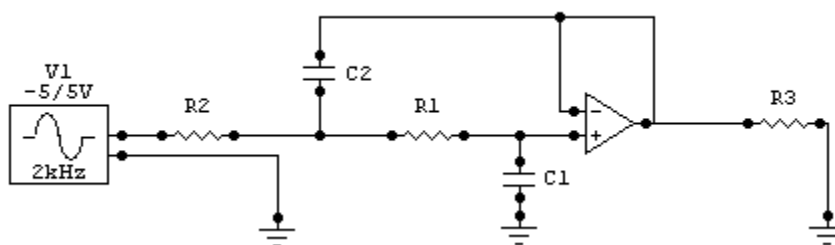


Fig. 6 - Butterworth Filter

The transfer function of a Butterworth filter is:

$$|H(j\omega)| = 1 / \sqrt{1 + (\omega / \omega_c)^{2n}}$$

where n is the order of the filter, ω is the frequency in question and ω_c is the corner frequency. In this case, we need to solve for filter order and cutoff frequency. To find the gain at a given frequency we can re-write the transfer function in the following way:

$$\text{gain}(\omega_x) = -10 \log(1 + (\omega_x / \omega_c)^{2n})$$

Waving our algebraic magic wand we can put this into the form:

$$g(\omega_x) = (\omega_x / \omega_c)^{2n} = \log^{-1}(-\text{gain}(\omega_x)/10) - 1$$

so if ω_s is the stopband frequency and ω_p is the frequency an octave below our cutoff:

$$\begin{aligned} g(\omega_s) &= \log^{-1}(13 \text{ dB} / 10) - 1 \\ &= 18.95 \end{aligned}$$

$$\begin{aligned} g(\omega_p) &= \log^{-1}(1 \text{ dB} / 10) - 1 \\ &= 0.26 \end{aligned}$$

which allows us to solve for n :

$$\begin{aligned} n &= \log(g(\omega_s) / g(\omega_p)) / 2 \log(\omega_s / \omega_p) \\ &= \log(18.95 / 0.26) / 2 \log(2000 / 500) \\ &= 1.86 / 1.20 \\ &= 1.55 \end{aligned}$$

which in this case we would round up to 2 since n must be an integer. Plug it into our $g(\omega_x)$ equation:

$$\begin{aligned} g(\omega_s) &= 18.95 = (2000 / \omega_c)^4 \\ &= 18.95^{1/4} = 2000 / \omega_c \\ \omega_c &= 2000 / 2.01 \end{aligned}$$

$$\omega_c = 995 \text{ Hz}$$

Let's call it 1000 Hz (what's 5 Hz between friends?). Knowing the order of our filter and the cutoff frequency, we can make some decisions about the actual circuit. The transfer function as it stands is not of much use to us, but we can substitute component values, making it much more useful [16]:

$$H(\omega) = R1 * R2 / s^2(C1 * C2) + sC1(R1 + R2) + R1 * R2$$

Refer to Figure 6 above for component numbers. We can make our lives much easier by setting all of the resistor values to 1Ω . In this way we can solve for just capacitor values:

$$H(\omega) = 1 / s^2(C1*C2) + 2sC1 + 1$$

We then need to determine our pole locations. Poles take the form $z1 = (-a + jb)$ and $z2 = (-a - jb)$. The roots of our transfer function are $(s+z1)(s+z2)$. This gives us roots of a quadratic equation: $(s - a - jb)$ and $(s - a + jb)$ which we multiply out to get:

$$s^2 + 2as + a^2 + jb^2$$

which takes the same form as the denominator of our transfer function above. [16] We go to our handy lookup tables for pole values. Since this is a 2-pole Butterworth, our poles are at $a = -0.707$ and $jb = +/- 0.707$. Plug this into the equation to get:

$$s^2 + 1.414s + 1$$

From this we can figure out that if our resistors are 1Ω , and we have a frequency of 1 rad/s , then $C1$ must be 0.707 Farads and $C2$ must be 1.414 Farads . To convert the radial frequency to Hz , we have to divide the capacitor values by 2π , which results in $C1 = 0.1125 \text{ F}$ and $C2 = 0.2252 \text{ F}$. Now we can make our resistors into whatever value we like, so long as we *divide* our capacitances by the same value. This means if we wish $R1$ and $R2$ to have values of $1 \text{ k}\Omega$, ($R * 1000$) our $C1 = 112.5 \mu\text{F}$ and $C2 = 225.2 \mu\text{F}$ ($C / 1000$). But our frequency is still 1 Hz , so to put this at our corner frequency ($f_c = 1\text{kHz}$) we divide the capacitors by 1000 again. Our circuit will end up with $R1$ and $R2 = 1 \text{ k}\Omega$, $C1 = 112.5\text{nF}$ and $C2 = 225.2\text{nF}$. By running a simulation in CircuitMaker with a $5V_p$ ($3.5V \text{ RMS}$) signal generator on the circuit in Figure 12, I measured RMS voltages of $3.4V$ at 500 Hz , $2.4V$ at 1 kHz and $0.85V$ at 2kHz . This corresponds to -0.3 dB , -3.0dB , and -12.3 dB respectively.

I also decided to try my hand at a 2nd-order low-pass Chebyshev using the method outlined in [16]. Chebyshev filters are defined in [13] as being "designed to have ripple in the passband, but steeper rolloff after the cutoff frequency." Where the Butterworth rotates the poles on a circle around the point of interest, the Chebyshev collapses this circle into an ellipse. This results in a steeper cutoff slope, but it also causes a sharp rise in the passband just before the dropoff point called an "overshoot". The flatter the ellipse, the greater the overshoot [11]. According to Smith: "As the ripple increases (bad), the roll-off becomes sharper (good). The Chebyshev response is an optimal tradoff between these two parameters." [25] (p.333) Interestingly enough, Smith goes

on to describe Butterworth filters as Chebyshev filters with the ripple parameter set to 0%. If we were to use higher-order Chebyshev filters, each individual pole would cause a corresponding individual rise in the passband resulting in a ripple in the frequency response.

Since the Chebyshev uses an elliptical arc around the origin, I used the poles specified in [16] for the first stage of a five-pole Chebyshev, which were $-0.2265 \pm j0.5918$. This gives us the quadratic formula:

$$\begin{aligned} s^2 + 2(0.2265)s + 0.0513 + 0.3502 \\ s^2 + 0.453s + 0.402 \end{aligned}$$

As before, it is easier to work with the resistor values set to 1 so that we can solve for capacitors, so we multiply each of the terms by $1 / 0.402$ resulting in the final form:

$$2.488s^2 + 1.127s + 1$$

Which means (referring again to our diagram and formula above for component numbers) $C1 * C2 = 2.488$ and $2 * C1 = 1.127$. This gives us capacitor values of $C1 = 0.5635F$ and $C2 = 4.42F$ for a low-pass filter with 1Ω resistors and a cutoff of 1rad/second. To use $1k\Omega$ resistors, we divide the capacitor values by 1000, and to convert to a cutoff frequency of 1000Hz, we divide them again by $2\pi * 1000Hz$. We now have $C1 = 90nF$ and $C2 = 704nF$.

Once again, I modelled the circuit in CircuitMaker to see its behaviour. I measured RMS voltages of 5.14V at 500 Hz, 1.87V at 1 kHz, and 0.43V at 2 kHz. This corresponds to +3.37 dB, -5.44 dB, -18.21 dB respectively. Note the rather large gain at one octave below the cutoff frequency. Also note that the cutoff frequency is slightly above its usual -6 dB, but also note the difference in gain one octave above the cutoff frequency when compared with the Butterworth filter.

There are lots of variations on this method of filter design (ie. different manifestations of the transfer function, using coefficient tables instead of pole locations, etc.) but the basic mechanics of it are the same.

It is common practice in both the analog and digital domains to design a higher-order filter by simply cascading a number of two-pole filters one after the other to achieve the desired response. However, you cannot simply cascade two identical filters one after the other. Depending on the order of the filter you are designing, the individual poles will be shifted around on the circle or the ellipse as more are added, always having a mirror reflection on either side of the σ axis.

All of this is fine and good but, after all, this paper is for a Digital Signal Theory

class, so why the concentration on analog design? It is not uncommon to first design a filter in the analog realm and then transfer it to the digital realm. There are different ways of doing this depending on the type of digital filter you are designing.

Digital Filters

Digital filters accomplish with math what an analog electronic filters accomplish with the laws of physics. A digital system is a discrete system, which is defined in [9] as "any system that accepts one or more discrete input signals x_n and produces one or more discrete output signals y_n in accordance with a set of operating rules." (p.595) The discrete input signal referred to here is a sequence of audio samples taking place over a range of time 0 through N. Each sample has a value that is defined at a specific point in time (n). Because we have captured and stored these values, we can manipulate them mathematically. If $y_n = x_n$, we have a unity gain filter: the output (y) at sample time "0" equals the input (x) at sample time "0." We could give our digital filter a gain coefficient "a" to scale the output: $y_n = a * x_n$. Likewise we could hold our input in some kind of buffer and delay its output by one sample: $y_n = x_{n-1}$

We can combine all of these ideas, making the output dependant on delayed inputs, each scaled by its own coefficient. A finite impulse response (FIR) filter uses such a scaled multi-tap delay structure:

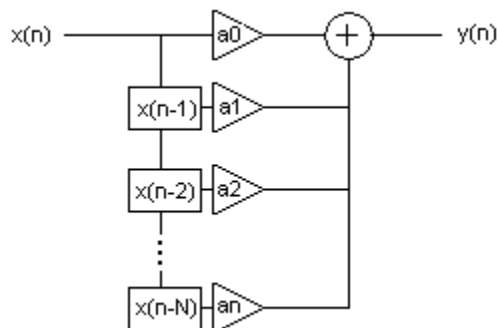


Fig. 7 - Typical FIR Structure

The frequency response of the filter is dependent on the number of delay elements and the corresponding set of coefficients. The transfer function of the FIR above takes the form:

$$y_n = a_0 * x_n + a_1 * x_{n-1} + a_2 * x_{n-2} + \dots + a_n * x_{(n-N)}$$

Dan Lavry explains the response of an FIR filter to an impulse: "The impulse 'travels'...against a 'fixed frame' of coefficients...the 'next' output sample is determined by shifting the signal-coefficient alignment by one and recomputing the sum of the products." [14] This impulse is multiplied by the coefficient at that particular tap resulting in a different value at each tap. If we were to collect the values and plot them on a graph of amplitude and time, we would see the impulse response of the filter. The impulse response of a filter is its signature. If

the impulse response of an analog filter is known, we could attempt to replicate it by manipulating the coefficients of the FIR.[8]

By taking the average of a certain number of past inputs, we would have a basic low-pass filter. Here is a simple FIR that I designed in Octave:

```
function y = myFIRfilter(input, srate, tapNumber, wetDryMix)

vectorLength = length(input) + tapNumber;
drySignal = [input; zeros(vectorLength - length(input), 1)];
drySignal1 = drySignal;
getRows = tapNumber;
taps = [zeros(vectorLength, 1)];
n=1;

    while (getRows > 0)
        delay = n;
        drySignal1 = drySignal1;
        currentTap = [zeros(delay, 1); drySignal1(1:1:(vectorLength - (delay)))];
        taps = taps .+ (currentTap * 1);
        n++;
        getRows--;
    endwhile

y = ((drySignal * (wetDryMix)) .+ (taps * (1-wetDryMix))) / (tapNumber + 1);
y = y/max(abs(y));
endfunction
```

A few weeks after designing this filter, I learned that there is actually a name for it: the Moving Average Filter [25]. In this design, the inputs are divided by the number of inputs giving a transfer function along the lines of: $y_n = (x_n + x_{n-1} + x_{n-2} + \dots + x_{n-N}) / N$. This is hardly an ideal low-pass filter, but it does do the job. I must admit, I do not yet know how to set the cutoff frequency on this filter, or what the actual slope is. The order of an FIR filter is measured by the number of taps that it uses in computing the output.

I tested this filter on several waveforms generated using the *siggen* function - sine wave, square wave, and triangle wave. It worked best on the triangle wave, turning it into a sine wave, thus proving that it was reducing the contribution of the high-frequency harmonics. The square wave would morph into a triangle wave. It did reduce the amplitude of higher-frequency sine waves. I need to experiment further with this design, as I am not sure if the normalization or the mixture of wet and dry signal should even be in this function. It could also easily be modified so that the user could enter a coefficient for each tap in the form of a vector (rather than having all the coefficients set to 1 as they are here), whether or not to take an average, how many inputs should be averaged, etc. That is left for future projects.

IIR filters make use of previous outputs, as well as previous inputs. Because they use feedback, IIR's are much more efficient and much more powerful than FIR's

$$y_n = x_n + y_{n-1}$$

[17] gives a good illustration of how this simple filter can equal or outperform an FIR design, because as time goes on, the IIR can represent an FIR of increasing order:

$$\begin{aligned} y_0 &= x_0 + y_{-1} && = x_0 \\ y_1 &= x_1 + y_0 && = x_1 + x_0 \\ y_2 &= x_2 + y_1 && = x_2 + x_1 + x_0 \\ y_3 &= x_3 + y_2 && = x_3 + x_2 + x_1 + x_0 \\ &\dots\text{etc.} \end{aligned}$$

The order of an IIR is based on the largest number of previous inputs and/or outputs. The above example is based on one previous output so it is a first order filter. An IIR usually has a short "feed forward" component - like a low-order FIR - and a feedback component. As with an FIR, we can assign coefficients to each tap - both feed forward and feedback. Usually, the coefficients for the outputs and feedback (y's) are represented by a "b" and the coefficients for the feed forward structure are represented by "a's" in the transfer function:

$$y_n = (a_0x_n + a_1x_{n-1} + a_2x_{n-2} - b_1y_{n-1} - b_2y_{n-2}) / b_0$$

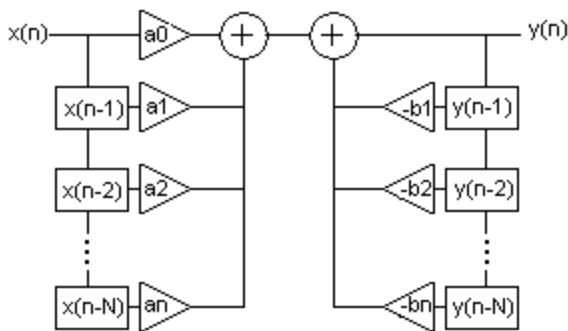


Fig. 14 - Basic IIR filter

There are several different ways to conceive and implement an IIR filter. To implement these architectures requires the introduction of the delay operator z^{-1} . This delay operator represents a one-sample delay. Instead of having to write y_{n-1} , x_{n-1} , y_{n-2} , etc., we can use z^{-1} instead. For instance:

$$\begin{aligned} z^{-1}x_n &= x_{n-1} \\ z^{-1}y_n &= y_{n-1} \end{aligned}$$

Applying this idea to the diagram in Figure 14, we see that we could actually combine the delay operators of the feed forward and feedback section of the filter, thus making the design itself less complicated:

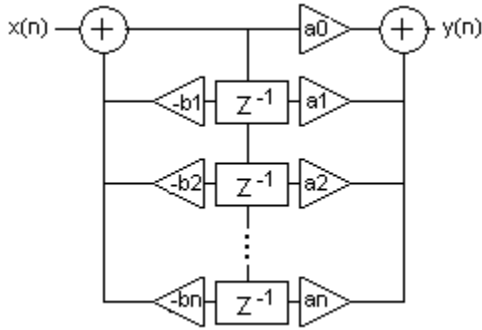


Fig. 15 - IIR filter - Direct Form II Structure

It also stands to reason that if z^{-1} represents a one-sample delay, then z^{-2} represents a two-sample delay ($z^{-2}x_n = x_{n-2}$). Using the delay operator, we can represent our design with the following equation:

$$Y(z) = a_0X(z) + a_1z^{-1}X(z) + a_2z^{-2}X(z) + \dots + a_Nz^{-N}X(z) - b_1z^{-1}Y(z) - b_2z^{-2}Y(z) - \dots - b_Nz^{-N}Y(z)$$

This is called a Z-transform. We can use it to find the response of our filter. Recall the basic transfer function:

$$\text{Gain}(f) = \text{output}(f) / \text{input}(f)$$

We can apply this here to say that $H(z) = Y(z) / X(z)$ which gives us the equation:

$$H(z) = [a_0 + a_1z^{-1} + a_2z^{-2} + \dots + a_Nz^{-N}] / [b_0 + b_1z^{-1} + b_2z^{-2} + \dots + b_Nz^{-N}]$$

since b_0 is taken to be 1, we multiply and divide by z^N and end up with the following:

$$H(z) = [a_0z^N + a_1z^{N-1} + a_2z^{N-2} + \dots + a_N] / [1 + b_1z^{N-1} + b_2z^{N-2} + \dots + b_N]$$

This brings us to another mathematical concept of filter design. The z-plane and the s-plane are closely related. Where the s-plane represents continuous-time (analog) filters, the z-plane is used to represent discrete-time (digital) filters. Every point on the imaginary axis ($j\omega$) in the s-plane has a corresponding point on the unit circle in the z-plane. This is important because it means that a filter designed in the analog realm can be transferred into the digital realm. An n -pole analog filter will have an equivalent n -pole IIR filter, and all that is needed to make one into the other is a bilinear transform.

I wanted to design a filter in Octave that would demonstrate this. My lack of math really hampered me in my efforts. I tried valiantly to transform my Butterworth into a digital IIR filter, but I found myself failing miserably. After about a week or

two of struggling with the algebra, I resorted to asking a colleague for help in working my way through transforming my s-domain transfer function into a z-domain transfer function. Dr. John Gordon of the Queensborough Community College Math and Computer Science Department was kind enough to show me the transformation from one to the other step by step guided by [26], without a specific frequency or sampling rate. After much algebra the results were:

$$a_0 = 1 / (\alpha^2 + 1.414\alpha + 1)$$

$$a_1 = 2 / (\alpha^2 + 1.414\alpha + 1)$$

$$a_3 = a_0$$

$$b_1 = (2 - 2\alpha^2) / (\alpha^2 + 1.414\alpha + 1)$$

$$b_2 = (1 - 1.414\alpha + \alpha^2) / (\alpha^2 + 1.414\alpha + 1)$$

where $\alpha = K / \omega_p$

K being some constant (either "1" or "2 / T" - I am not clear on that) and ω_p being the warped analog cutoff frequency [26]. Here was one source of great confusion: no matter how many articles I read about the bilinear transform, I would find yet another way of "warping" the analog cutoff frequency of a filter. This seemed to prove the greatest hurdle for me as I could not figure out mathematically how this translated from one method to another. Even when using the same warping method with the transformation, I was unsuccessful in predicting the filter's response.

In any case, I believe that the coefficients Dr. Gordon came up with are correct, but utilizing his coefficients with my cutoff frequency, the filter did not work in the manner expected. Faced with a deadline fast approaching, I resorted to the "coward's way out" and went to a coefficient table found in [25]. The only sense I could make of this table where f_c was listed as a range of numbers from 0.01 to 0.45 was that:

$$f_c = f_d / \text{srate}$$

where f_d is the frequency of the sine wave generated by *siggen*. Since it seemed to work I decided not to question any further at this time and simply move on with implementing my filter.

Since 1000 Hz at 48 kHz sampling rate would translate to 1 Hz at 48 Hz sampling rate, I set my cutoff frequency for 3.6kHz with a 48 kHz sampling rate, which should translate to 3.6 Hz at 48 Hz sampling rate. Using this frequency, I would be able to demonstrate frequencies both above and below the cutoff with some reliability in Octave. On the coefficient table in [25], this seemed to correspond to 0.075. So being the coward that I am, I plugged the coefficients into my already half-implemented filter:

```

function y = myIIR348(input, srate, durSecs)

N=srate*durSecs;

x=[input; zeros(N-length(input), 1)];
y=[x(1); x(2); zeros(N-2, 1)];

for n = 3:N

y(n) = .03869430*x(n) + .07738860*x(n-1) + .03869430*x(n-2) +
1.392667*y(n-1) - 0.5474446*y(n-2);

endfor
endfunction

```

2 Hz should be approximately 1 octave below the cutoff, 4 Hz should be close to the cutoff, and the octave above should fall around 7 Hz. Using the *siggen* function, I generated these sine waves as xsw2, xsw4 and xsw7, all with peak amplitude values of 1.0. Looking through the vectors generated by these functions after they were passed through my IIR filter, xsw2 still had peaks around 0.95, xsw4 had peak values at about +/- 0.6, and xsw7 had peaks around +/- 0.2. This would correspond to approximately -0.5dB, -4.5 dB, and -14 dB respectively.

I passed a 4 Hz square wave through the filter and ended up with a 4 Hz sine wave with reduced amplitude corresponding to that of xsw4. I also tried a 2 Hz square wave and again, a sine wave was the result at the output, though it looked just a bit lopsided, almost as though the rate of change was not even throughout the curve.

Further study is needed on my part - especially in the area of higher math - before I can truly begin to understand the world of digital filter design. However, I was pleased that I was at least able to implement the two basic types of filters and see that the theory actually worked.

Apodizing Low-Pass Filters

Smith [25] mentions that there is a limit to the amount of poles that can be used in a single stage low-pass filter, with the smallest number of poles being on the extremes - DC and the Nyquist limit. The maximum number of poles that can be used at about the Nyquist frequency is 4 - which yields an attenuation slope of about -24 dB per octave. "The filter's performance will start to degrade as this limit is approached; the step response will show more overshoot, the stopband attenuation will be poor, and the frequency response will have excessive ripple." [25] (p.339) The ramifications of having your sampling rate just barely adequate to cover your desired bandwidth are clear. A cutoff slope that is steep enough to prevent aliasing will cause artifacts in the passband - be it ripple or phase smearing from having to cascade multiple stages. This is the case with digital audio sampled at 44.1 kHz.

This brings me to the point where I actually began my quest to understand filter-design - an AES Journal article that sparked my interest in low-pass filters entitled *Antialias Filters and System Transient Response at High Sample Rates* by Peter Craven. In the introduction of the paper he makes the statement, "The reported improvement [of high sampling rates] has been attributed to the ear's sensitivity to "time smear" produced by the antialias and reconstruction filters used in digital recording...and especially by the "brickwall" filters typically used at 48 or 44.1 kHz." He goes on to say that "a single apodizing filter can be used to control the time smear of a complete recording and reproducing chain." [5] (p216)

Apodization is the act of removing or smoothing "a sharp discontinuity in a mathematical function" or an electrical signal. [27] Brickwall antialias filters, by their very definition, typically have sharp band edges. Ricardo Losada [28] makes the point that "the impulse response required to implement the ideal lowpass filter is infinitely long." In other words, the sharper the band edge, the longer the filter will ring in response to an impulse. The longer the ring, the worse the disparity in group delay. Craven states that "it is possible to compensate a filter that gives a nonuniform group delay by following it with another filter having the complementary delay." (p219) He demonstrates that using this technique of apodization, not only can you remove the ringing of a single filter, you can also remove the ringing of every filter in the chain since "Mathematically, the frequency response of the cascade of filters is the product of the (complex) frequency responses of all the filters in the cascade..." (p.219) the problem being that you need to know the properties of all of the filters used in the chain before the apodizing filter. In the placement strategy section, he mentions that it is preferred to have only one apodizing filter in the chain, although it would probably "not be a disaster to cascade two apodizing filters." (p.236) He concludes that it seems most logical to encourage mastering engineers to experiment with this technique, and perhaps at some point, manufacturers of playback equipment might include apodizing filters as an option.

Unfortunately, much of Craven's article is beyond the scope of this paper, but I hope to study this topic further as well. Regardless, it is clear that a higher sampling rate means that the antialiasing filter need not have such a sharp cutoff, and so the overall passband response can benefit from simply having some breathing room to tailor the response of the filter. Since the rolloff can be more gradual, the cutoff slope can be more smooth and so there is less of a problem with nonuniform group delay.

Conclusion

I started this paper knowing only that you could build a low-pass using a resistor and a capacitor. I never dreamed how complicated the process of designing filters actually was. In this light, I have a newfound respect for the art and

science of filter design. I hope that I have the chance to further my knowledge of this topic, but even so I have already answered a number of my own questions about the debate of converter design and the effect of higher sampling rates, and a new perspective on the benefits and tradeoffs involved.

Sources

1. Grob, Bernard. *Basic Electronics*, Eighth Edition. Glencoe/McGraw-Hill, New York. 1999
2. Gibilisco, Stan. *Teach Yourself Electricity and Electronics*, Third Edition. McGraw-Hill, New York. 2002
3. Lavry, Dan. *Sampling, Oversampling, Imaging and Aliasing - a basic tutorial*. Lavry Engineering 1997
4. Bohn, Dennis A. *Signal Processing Fundamentals*. RaneNote 134 revised 2004
<http://www.rane.com/note134.html>
5. Craven, Peter G. *Antialias Filters and System Tansient Response at High Sample Rates*. Journal of AES, Vol. 52, No. 3, March 2004
6. http://en.wikipedia.org/wiki/Low-pass_filter. Nov. 2005
7. PSW Recording Forums: Dan Lavry -> current lags voltage question. Oct. 2004
<http://recfroums.prosoundweb.com/index.php/t/2138/0>
8. notes from: *Digital Filters and Filter Design: A Tutorial*. 119th AES Convention, Tutorial Seminar 21. October 10, 2005. Presented by James Johnston (Microsoft), Bob Adams (Analog Devices), and Jayant Datta (Discrete Laboratories).
9. Pohlmann, Ken C. *Principals of Digital Audio*, Fourth Edition. McGraw-Hill, New York. 2000
10. Stark, Peter A. *Operational Amplifiers*. Peter Stark. 2004
11. Stark, Peter A. *High-Pass and Low-Pass Filters*. Peter Stark. 2004
12. Losmandy, B.J. *Operation Amplifier Applications For Audio Systems*. Journal of the AES, Vol. 17, No.1, January 1969
13. Karki, Jim. *Active Low-Pass Filter Design*. Texas Instruments. October 2000
14. Lavry, Dan. *Understanding FIR (Finite Impulse Response) Filters - An Intuitive Approach*. Lavry Engineering. 1997

15. Baraniuk, Richard. *Butterworth Filters*. version 2.10: 2005/07/18
<http://cnx.rice.edu/content/m10127/latest/>
16. *Analog Filter Design Demystified*. Dallas Semiconductor Application note 1795. 2002
http://www.maxim-ic.com/appnotes.cfm/appnote_number/1795
17. *Introduction to Digital Filters*. Author Unknown
<http://www.dsptutor.freeuk.com/dfilt1.htm>
18. Lavry, Dan. *Understanding IIR (Infinite Impulse Response) Filters - An Intuitive Approach*. Lavry Engineering. 1997
19. Chassaing, Rulph. *Digital Signal Processing with C and the TMS 320C30*. John Wiley & Sons Inc. New York 1992
20. Smith, Julius O. III. *Introduction to Digital Filters with Audio Applications*. Julius O. Smith III. 2005
21. http://en.wikipedia.org/wiki/Bilinear_transform
22. Stark, Peter A. *Band-Pass Filters and Resonance*. Peter Stark. 2004
23. *A Filter Primer*. Dallas Semiconductor Application note 733. 2001
http://www.maxim-ic.com/appnotes.cfm/appnote_number/733
24. *All-Pole IIR Filters*.
<http://unicorn.us.com/alex/allpolefilters.html>
25. Smith, Steven W. *The Scientist and Engineer's Guide to Digital Signal Processing*. California Technical Publishing. 1999
www.dspguide.com
26. PSW Recording Forums: Dan Lavry -> So I have my coefficients - now what...Dec. 2005
<http://recfroums.prosoundweb.com/index.php/t/8389/0>
27. <http://en.wikipedia.org/wiki/Apodization>
28. Losada, Ricardo A. *Design Finite Impulse Response Digital Filters*.
Microwaves & RF Jan. 2004
<http://www.mwrf.com/Articles/Print.cfm?ArticleID=7229>

Special Thanks to: Professor Peter Stark and Dr. John Gordon of Queensborough Community College

Appendix - Waveform Graphics