Linear Filters

Introduction

A *linear filter* is a circuit with an output $\mathbf{V}_{\text{out}}$ that is a linear function of its input $\mathbf{V}_{\text{in}}$ and that is intended to remove unwanted parts of a signal. Such filters are often described by specifying how they modify the amplitude and phase of each frequency component of the signal with a **transfer function**

$$\mathbf{G}(\omega) = \frac{\mathbf{V}_{\text{out}}(\omega)}{\mathbf{V}_{\text{in}}(\omega)} = \frac{\mathbf{a}_0 + \mathbf{a}_1\omega + \mathbf{a}_2\omega^2 + \ldots}{\mathbf{b}_0 + \mathbf{b}_1\omega + \mathbf{b}_2\omega^2 + \ldots}. \tag{1}$$

This quotient of two polynomials is known as a Padé function and is usually expressed as the product of its complex poles and zeros, *e.g.*

$$\mathbf{G}(\omega) = \frac{(\boldsymbol{\omega}_{\text{z0}} + j\omega) \times (\boldsymbol{\omega}_{\text{z1}} + j\omega) \times \ldots}{(\boldsymbol{\omega}_{\text{p0}} + j\omega) \times (\boldsymbol{\omega}_{\text{p1}} + j\omega) \times \ldots} \tag{2}$$

Much cryptic engineering-jargon arises from this convention. For example a ‘single-pole op-amp’ is one with an open-loop gain accurately described by

$$\mathbf{A}(\omega) = \frac{a_0}{\omega_0 + j\omega} \tag{3}$$

which has a single complex pole at $\omega = j\omega_0$.

In principle, designing a filter is a straightforward matter – one uses a computer and/or creates a chain of stages until the desired transfer function is obtained. However, non-trivial designs often do not work well; they tend to require extremely precise component values that are not available in practice.

There are many of types of filter; the most commonly encountered include: Butterworth (maximum flatness in pass-band), Chebyshev (sharpest transition from pass-band to stop-band), Bessel (best linear phase *vs* frequency response).

General Remarks

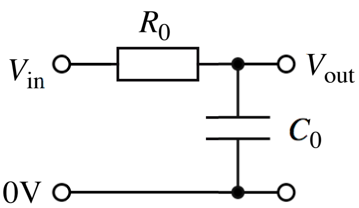
In the context of test and measurement systems, filters irreversibly remove information from the signal and should be used sparingly, particularly if it would be possible to post-process signals digitally. Cases where analogy filtering is necessary include: preventing overloading of the signal channel by interference, and preventing 'aliasing' if the signal is to be digitized. Filter designs are normally very sensitive to component tolerances and often require expensive high-precision components and very careful construction in order to work correctly.

Passive Filters

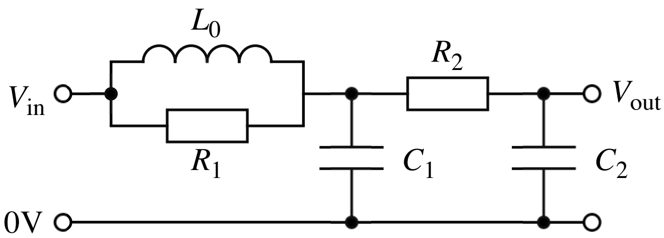
Passive filters are constructed entirely from R, L and C elements and therefore require no power supply. The most common is the low-pass first-order RC-filter (circuit 1) which has a single-pole transfer function

$$\frac{\mathbf{V}_{\text{out}}}{\mathbf{V}_{\text{in}}} = \frac{1}{1 + j\omega C_0 R_0}, \quad \left|\frac{\mathbf{V}_{\text{out}}}{\mathbf{V}_{\text{in}}}\right| = \frac{1}{\left(1 + \omega^2 C_0^2 R_0^2\right)^{1/2}} \quad \text{and} \quad \phi = \arctan\left(-\omega C_0 R_0\right) \qquad \text{(4a–c)}$$

which attenuates at a rate of 6 dB per octave above its corner at $f_t = 1/2\pi R_0 C_0$.



Circuit 1.



Circuit 2.

Circuit 2 uses an inductor to achieve a rapid transition between the **pass-band** (*i.e.* the region of constant amplitude gain) below f_t and the **stop-band** above it where the attenuation is 18 dB per octave. Calculating the transfer function by hand is a laborious task

$$\frac{\mathbf{V}_{\text{out}}}{\mathbf{V}_{\text{in}}} = \frac{\alpha + \beta j\omega}{\alpha + Bj\omega - C\omega^2 - Dj\omega^3} \qquad \begin{array}{c} \alpha = R_1 R_2 \quad \beta = L_0 R_2 \quad \mathrm{B} = \mathrm{C}_2 R_1 R_2^2 + L_0 R_2 \\ C = L_0 R_2\left[C_2\left(R_1 + R_2\right) + \mathrm{C}_1 R_1\right] \quad D = \mathrm{C}_1 C_2 L_0 R_1 R_2^2 \end{array} \qquad (5)$$

A major problem with passive filters is that they normally require the use of inductors. This makes them desirable be inductors are normally: bulky, fragile, sensitive to electromagnetic interference, and expensive.

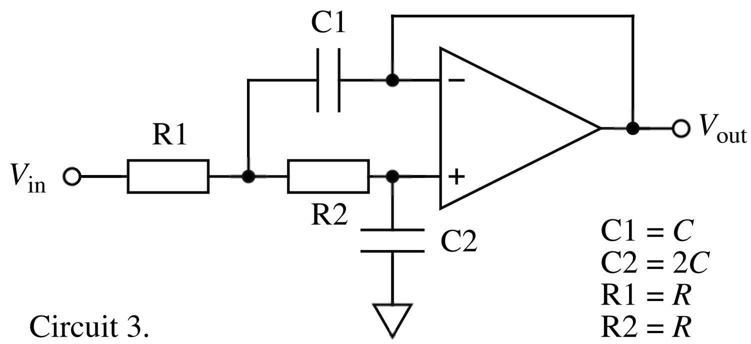
Active Filters

Filter performance can be enhanced if active elements are used. For example, the finite impedance of one stage in a passive design will affect the behaviour of its neighbours, but an op-amp in a unity-gain configuration can prevent this problem by isolating the stages.

Linear Phase Filters

The frequency-domain phase-shift of a filter corresponds to a delay in the time-domain. If the phase shift of a filter increases linearly with frequency, *i.e.* $\phi(\omega) = t_d\omega$, then every frequency component will experience the same delay t_d. As a result, signals that pass through the system will be delayed, but not distorted by distortion. This linear-phase property is often important in signal processing applications and Bessel filters are optimized to provide it.

Example – Sallen and Key Filter



Circuit 3.

This design for a second-order filter works well in practice. Circuit 3 is the low-pass version, which the transfer function

$$\frac{\mathbf{V}_{out}}{\mathbf{V}_{in}} = \frac{1}{1+\sqrt{2}(j\omega/\omega_c)-(\omega/\omega_c)^2} \quad \text{where} \quad \omega_c = \sqrt{\frac{\pi}{2CR}} \tag{6}$$

The high-pass version, with transfer function

$$\frac{\mathbf{V}_{out}}{\mathbf{V}_{in}} = \frac{1}{1-\sqrt{2}(j\omega/\omega_c)-(\omega/\omega_c)^2} \quad \text{where} \quad \omega_c = \sqrt{\frac{\pi}{2CR}} \tag{7}$$

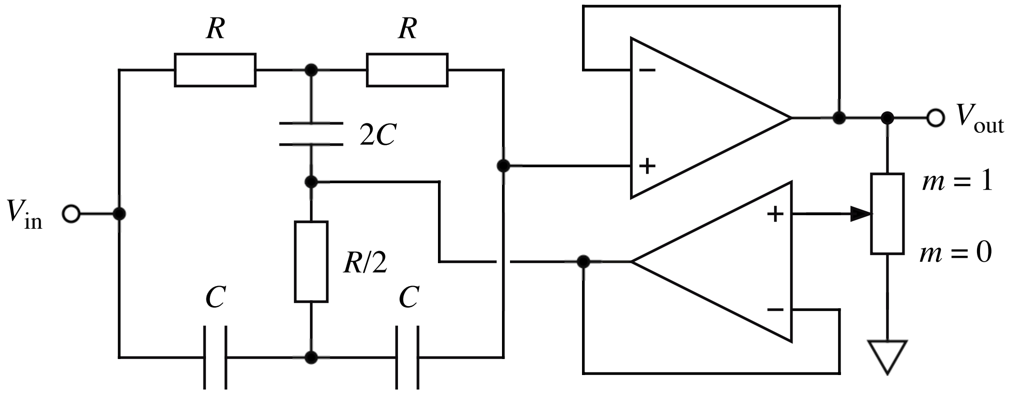
is made by substituting: $R \to C; \quad 2C \to R; \quad C \to 2R$.

Example – Notch Filter

Notch filters pass (or stop) a narrow range of frequencies. They are particularly useful for excluding mains frequency interference. The transfer function for circuit 4:

$$\frac{\mathbf{V}_{out}}{\mathbf{V}_{in}} = \frac{Q\left(j\omega^2-\omega_0^2\right)}{\omega\omega_0 + jQ\omega^2 - \omega_0^2} \quad \text{where} \quad \omega_0 = \frac{1}{CR} \quad \text{and} \quad Q = \frac{1}{4(1-m)} \tag{8}$$

is modified by adjusting the potentiometer. Q-factors of up to about 100 are achievable with this type of circuit, but much higher values are possible by using an entirely different approach based on a lock-in amplifier



Circuit 4.

Example – Spice Analysis of Poles and Zeros

To illustrate using of Spice 3 to analyse filters, consider Circuit 2 above, with component values:

$$R_1 = 15\ \text{k}\Omega \quad C_1 = 15.9\ \text{nF} \quad L_0 = 1.59\ \text{H}$$
$$R_2 = 100\ \text{k}\Omega \quad C_2 = 1.59\ \text{nF}.$$

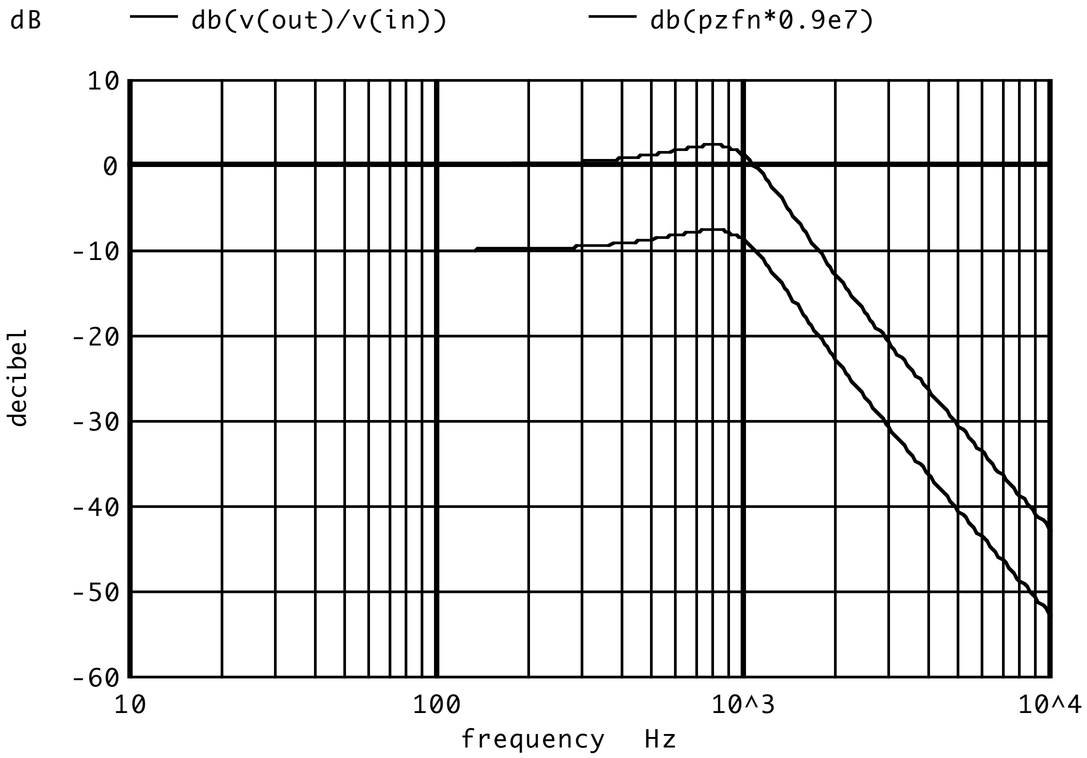
The source file (Appendix 1) contains the netlist and runs a 'PZ' analysis, the results are:

```
MacSpice 1 -> source Primary:Users:cdhw:Desktop:Filters:CW100322-01.CIR

Circuit: PHY3128 Filter circuit analysis example CW100322-01.CIR

pole(1) = (-6.81684e+03, 0.00000e+00 )
pole(2) = (-2.29688e+03, 5.58737e+03 )
pole(3) = (-2.29688e+03,-5.58737e+03 )
zero(1) = (-8.80503e+03, 0.00000e+00 )
MacSpice 2 ->
```

These are used to construct plot the transfer function

$$\mathbf{G}(\omega) = \text{constant} \times \frac{(-\texttt{zero}(1) + j\omega)}{(-\texttt{pole}(1) + j\omega) \times (-\texttt{pole}(2) + j\omega) \times (-\texttt{pole}(3) + j\omega)}, \tag{9}$$

which agrees with the frequency response calculated by an AC analysis.


dB
db(v(out)/v(in))
db(pzfn*0.9e7)
10
0
-10
-20
-30
-40
-50
-60
decibel
10
100
10^3
10^4
frequency Hz

Appendix

```
PHY3128 Filter circuit analysis example CW100322-01.CIR
**** CIRCUIT TOPOLOGY DEFINITION SECTION
R1 in X 14K
R2 X out 100K
C1 X 0 15.9n
C2 out 0 1.59n
L0 in X 1.59H
Vin in 0 DC 0.0 AC 1.0 0.0 PULSE( 0.0 1.0 1ms )
****  COMMANDS SECTION
.control
* Create plot containing poles and zeros, then print them
pz in 0 out 0  vol pz
print all
* Create plot containg ac analysis
ac  dec  100  10Hz  10kHz
* Plot the pole-zero transfer function
* First, create the s (complex frequency) vector
let s = i*frequency*2*pi
* Evaluate the pole-zero function. The current plot is 'ac' so
* the 'pz.' prefix is needed to evaluate the poles and zeros
* which are not in the current plot.
let pzfn = (1,0)*unitvec(length(s))
foreach j 1
       let pzfn = pzfn * ((-pz.zero($j)+s)/(-pz.pole($j)+s))
end
foreach j 2 3
       let pzfn = pzfn / (-pz.pole($j)+s)
end
* Display the results. Omit the constant factor from the pz plot
* for clarity
plot db(v(out)/v(in)) db(pzfn*0.9e7)
*plot phase(v(out)/v(in)) phase(pzfn)
.endc
.END
```

Supplementary Reading

Amplification – Storey (1998) §3.6–3.7 pp. 66–83 / (2006) §3.6–3.9 pp. 61–82.

Positive Feedback – Storey (1998) §8.1–8.2 pp. 312–323 / (2006) §11.1–11.2 pp. 350–356.