CHAPTER CONTENTS

10.0 Digital Signal Processing

Objectives

This section will:

- Review some basic digital filter theory
- Examine FIR filters
- Examine IIR filters
- Show how to design a digital filter using the impulse response
- Introduce convolution

http://www.dsptutor.freeuk.com/index.htm

http://www.bores.com/courses/intro/index.htm

http://www.nitehawk.com/rasmit/dsp50.html

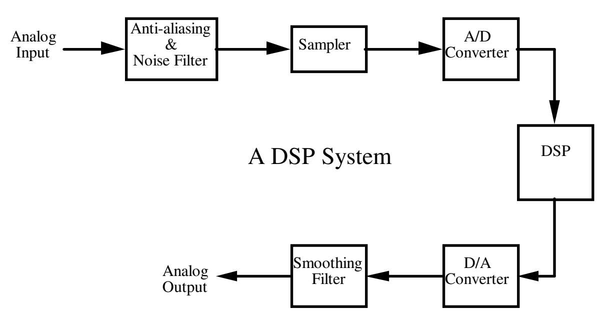
The first demonstration of a simple DSP device was in the 1930's, but the first commercial programmable device didn't come on the market until 1979. Since then DSP has revolutionized communications systems design and applications.

DSP has several significant advantages over traditional analog technology:

- Size and cost of digital ICs is falling
- Compatible with other digital technologies: transmission, switching, computers
- Can be programmable
- Stable and accurate
- Increased functionality

There are however some disadvantages. Since DSPs can perform more complex functions than other techniques, circuit designers need a good grasp of mathematics. Unfortunately, there are relatively few user-friendly development and debugging tools.

A Simple DSP System



A DSP System

From the above diagram, it should be evident that a telephony PCM codec is a DSP system.

In order to run real time applications, the DSP processor must be able to execute all of its functions within one sampling period of the input filters. Some applications, such as real time video processing are beyond the state of current technology. However, there are slow scan or stop motion video applications, which are receiving a great deal of attention.

Applications can be categorized by the time it takes to produce a useful result. Real time applications such as telephony, require the processing to take milliseconds or less. Near real-time applications may take seconds to respond, while others may take minutes, hours or even days.

Run Time	Application
Real Time	Telephony [codecs, cellular phones] Consumer audio [CDs, DAT, toys] Medical [ECG] Multimedia [video compression] Radar signal processing
Near Real Time	Adaptive equalizers Medical [CAT scan] Cruse missile guidance Voice & pattern recognition [machine vision]
Long Time	Satellite & deep space imaging

10.1 Digital Filters

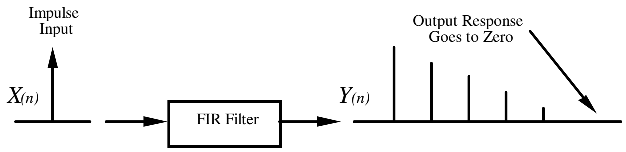
http://www-users.cs.york.ac.uk/~fisher/mkfilter/

Digital filters may be realized in hardware, software, or more often in firmware. Except for the most complex applications, they tend to be more expensive than analog filters, but they may be able to do things that analog filters cannot. Digital filters can be classified according to a wide range of characteristics, not all of which are mutually exclusive.

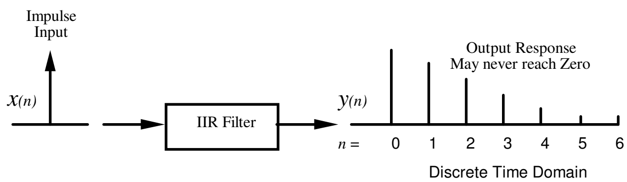
Classification	Examples
Impulse Response	FIR IIR
Topology	Serial or cascade Parallel
Cause	Non-Causal Causal
Feedback	Non-Recursive Recursive
Order	First, second, etc.

Analog filters are most often categorized by their frequency response, while digital filters are generally categorized by their time response, specifically, their impulse response.

When a FIR[†] filter is excited by an impulse, it generates a finite number of output values. This means that the output eventually decays to zero when the input is removed.



When an IIR[†] filter is excited by an impulse, it produces an infinite number of output values. This means that the output may not decay to zero when the input is removed.



This division is based on the circuit response to an impulse and is quite mathematical in nature. A more familiar division, for analog filter designers, is whether the filter has any feedback elements. FIR filters don't.

10.1.1 Difference Equations

An nth order linear difference equation with constant coefficients can be written as:

$$\begin{aligned} y(k) + b_1 y(k-1) + b_2 y(k-2) + \cdots + b_n y(k-n) \\ = a_0 x(k) + a_1 x(k-1) + a_2 x(k-2) + \cdots + a_m x(k-m) \end{aligned}$$

The difference equation is normalized by setting b_0 equal to one [$b_0 = 1$]. In closed form the equation when solved for the output $y(k)$ is written:

$$y(k) = \underbrace{\sum_{j=0}^{m} a_j x(k-j)}_{\text{Feed Forward}} - \underbrace{\sum_{j=1}^{n} b_j y(k-j)}_{\text{Feedback}}$$

These terms can be interpreted as follows:

$b_j\, y(k - j)$ is the output at any given instant in time

$a_j\, x(k - j)$ is the input at any given instant in time

The coefficient a and b represent the amplitudes

The notation $(k - j)$ represents the next sampling instant. This can also be interpreted as a delay equal to the sample period.

† Finite Impulse Response

† Infinite Impulse Response

In the most general case, there is no need to assume that there will be an equal number of *a* and *b* coefficients.

10.1.2 Z Transforms

Another notation, which is used to describe a unit delay, is z^{-1}. This notation forms the bases of Z or discrete time transforms. Most texts use small letters to describe signals in the discrete time domain, and capital letters to describe them in the Z domain.

$$y(k) \Leftrightarrow Y(z)$$
$$y(k-n) \Leftrightarrow Y(z)z^{-n}$$

The Z transform of the open form difference equation is:

$$\begin{aligned}&Y(z)+b_1Y(z)z^{-1}+b_2Y(z)z^{-2}+\cdots b_kY(z)z^{-k}\\&=a_0X(z)+a_1X(z)z^{-1}+a_2X(z)z^{-2}+\cdots+a_kX(z)z^{-k}\end{aligned}$$

The closed forms are given by:

$$Y(z)+\sum_{j=1}^{n}b_jY(z)z^{-j}=\sum_{j=0}^{m}a_jX(z)z^{-j}$$

$$Y(z)=\underbrace{\sum_{j=0}^{m}a_jX(z)z^{-j}}_{\text{Feed Forward}}-\underbrace{\sum_{j=1}^{n}b_jY(z)z^{-j}}_{\text{Feedback}}$$

Solving for the transfer function:

$$Y(z)\left(1+\sum_{j=1}^{n}b_jz^{-j}\right)=\sum_{j=0}^{m}a_jX(z)z^{-j}$$

$$\frac{Y(z)}{X(z)}=H(z)=\frac{\sum_{j=0}^{m}a_jz^{-j}}{1+\sum_{j=1}^{n}b_jz^{-j}}=\frac{a_0+a_1z^{-1}+a_2z^{-2}+\cdots a_mz^{-m}}{1+b_1z^{-1}+b_2z^{-2}+\cdots b_nz^{-n}}$$

Z transformed expressions in this form can be directly implemented as digital filters. Terms in the numerator correspond to zeros and those in the denominator to poles.

For the sake of analysis, the sampling rate can be normalized with respect to the incoming analog signal. Consequently, digital filter coefficients are functions of normalized frequencies and only the number of coefficients in a given period is important. Thus by changing the sampling rate in a digital filter, its frequency response characteristics can be shifted, thus supporting zooming and multirate filtering.

10.1.3 FIR Filters

The general expression for a FIR filter output is:

$$y(k) = \sum_{j=0}^{m} a_j x(k-j)$$

or

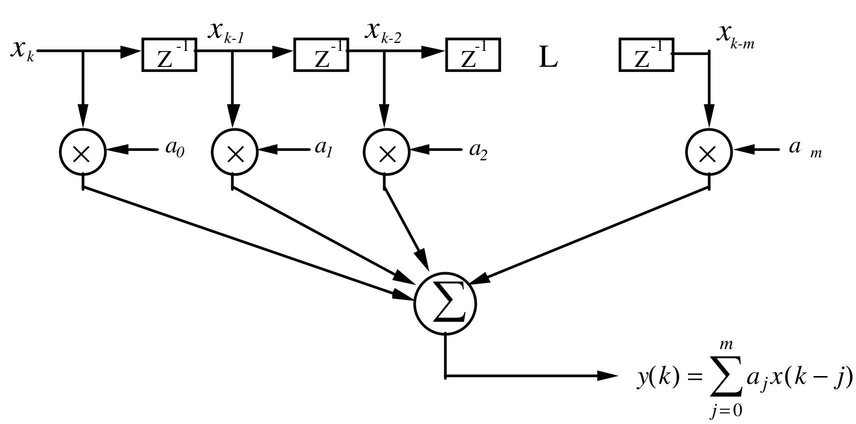
$$Y(z) = \sum_{j=0}^{m} a_j X(z) Z^{-j}$$

This filter does not employ feedback in the traditional sense, does not contain any poles and cannot oscillate. When an impulse is applied at the input, there are a finite number of output values. It is also known as a non-recursive filter. Some of its chief characteristics are:

- High stability
- Low round-off noise
- Can have linear phase
- Can be adaptive
- Use more components than IIR filters
- May be easy to understand, design, and implement

10.1.3.1 Direct Form

The direct form is the simplest of the FIR filters. It has no poles, and is stable.



The series Z^{-1} elements are often implemented as shift registers or memory locations. The balance of the circuit is a multiplying accumulator.

A sort of feedback can be incorporated to make this filter adaptive. The output can be compared to a reference pattern and used to generate an error signal. This is can be used to modify the filter coefficients dynamically. An example of such a device is the zero forcing equalizer. Note that using coefficient feedback does not convert this FIR filter into an IIR filter.

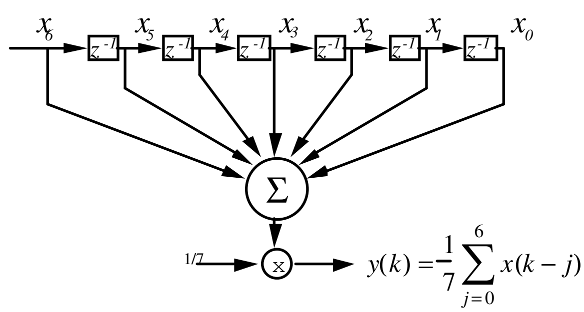
Digital filters are often designed by converting either an existing analog circuit of transfer function into the time or *Z* domain. Alternatively, one may start from the desired passband. Starting from an analog circuit is somewhat difficult, since analog filters have poles whereas a FIR filter does not.

10.1.3.2 Moving Average Filter

The RC integrator is a simple low pass filter, which requires recursive techniques to be implemented digitally. However, there are similar functions that can be performed by non-recursive methods. One of these is a moving average filter, which is a type of integrator or LPF. To average *n* events or samples, *n* - 1 delays and a multiplier is required. Since each coefficient in this example is the same, only one multiplier need be used. The multiplier coefficient is set to *1/n*.

Example 10-1: A 7-day moving average FIR filter.

Since all of the coefficients in this filter are the same, the multiplier can be places after the summer. Thus only one multiplication need be performed.



This device might, sample the maximum temperature for each day of the week and calculate the average over the past 7 days. Alternatively, it may be clocked at a high rate and used to reduce the effects of some random noise in a signal.

If the input is set to 1, the transfer function of this filter can easily be written in the *Z* domain:

$$H(z) = \frac{1}{7}\sum_{n=0}^{6} z^{n}$$

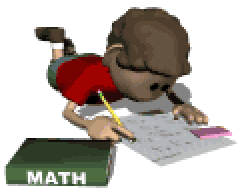
From math tables, we note that this result can be transformed into the *S* domain by the relation:

$$z^{-n} \Leftrightarrow e^{-nTs}$$

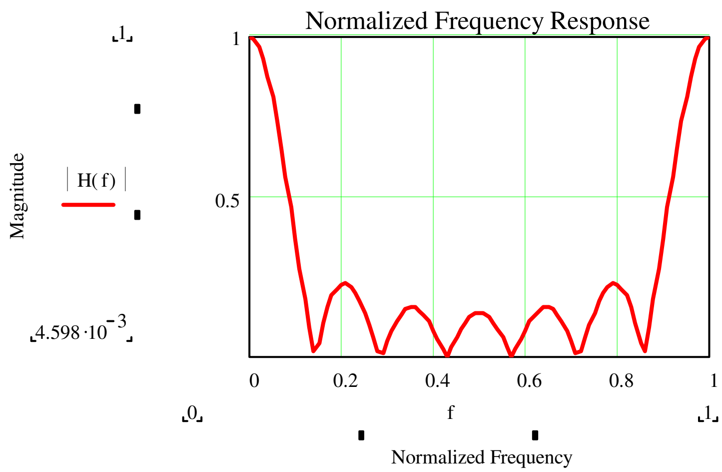
Consequently:

$$H(s) = \frac{1}{7}\sum_{n=0}^{6} e^{-nTs}$$

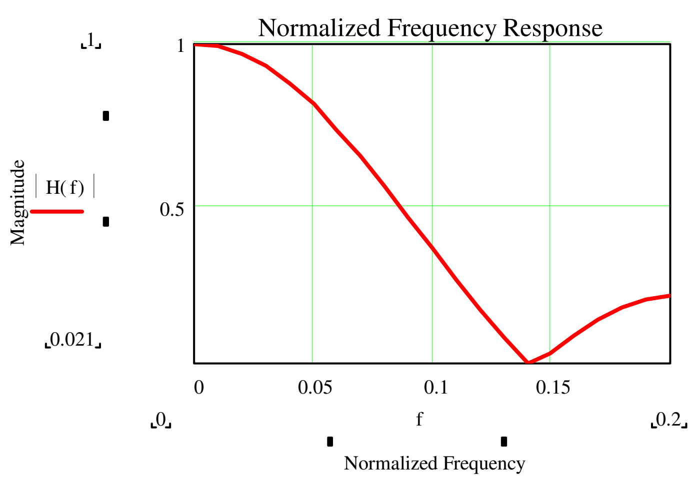
From here it is possible to obtain the frequency response by letting $s = j\omega$ and plotting the magnitude of the result over the interval $0 \le \omega \le \frac{2\pi}{T}$. The result is a low pass filter as might be expected.



MathCAD Moving Average Simulation



The low pass characteristic is more readily observed when zooming in on the response.



10.1.4 Impulse Response Design

Ideal low pass filters are impossible to fabricate, but a reasonable approximation can be made for some applications.

To design this type of filter:

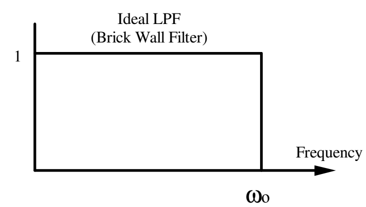
- Determine the desired frequency response
- Calculate the impulse time response by taking the real part of the inverse Fourier transform of the frequency response

$$f(t)=\frac{1}{2\pi}\int_{-\infty}^{\infty}F(j\omega)e^{j\omega t}d\omega$$
$$=\frac{1}{2\pi}\int_{-\infty}^{\infty}F(j\omega)\cos(\omega t)d\omega+\frac{1}{2\pi}j\int_{-\infty}^{\infty}F(j\omega)\sin(\omega t)d\omega$$

- Select the desired number of time delay taps [the more taps, the more faithful the reproduction]
- Use the calculated values as the FIR filter coefficients

Example 10-2: an ideal low pass filter

The passband response resembles:



The impulse response is found by taking the inverse Fourier transform.

$$f(t)=\frac{1}{2\pi}\int_{0}^{\omega_o}1\times e^{j\omega t}d\omega=\frac{1}{2\pi}\frac{1}{jt}e^{j\omega t}\Big|_{0}^{\omega_o}$$
$$=\frac{1}{2\pi\ jt}\left(e^{j\omega_o t}-1\right)$$

This can be simplified by applying Euler's identity:

$$e^{j\omega t}=\cos(\omega t)+j\sin(\omega t)$$

Therefore:

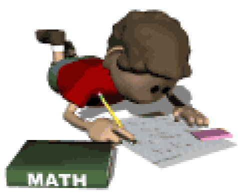
$$f(t)=\frac{1}{2\pi\, t\, j}\left(\cos(\omega_o t)+j\sin(\omega_o t)-1\right)$$
$$=-\frac{j}{2\pi t}\cos(\omega_o t)+\frac{1}{2\pi t}\sin(\omega_o t)+\frac{j}{2\pi t}$$

The real part of the result is given by:

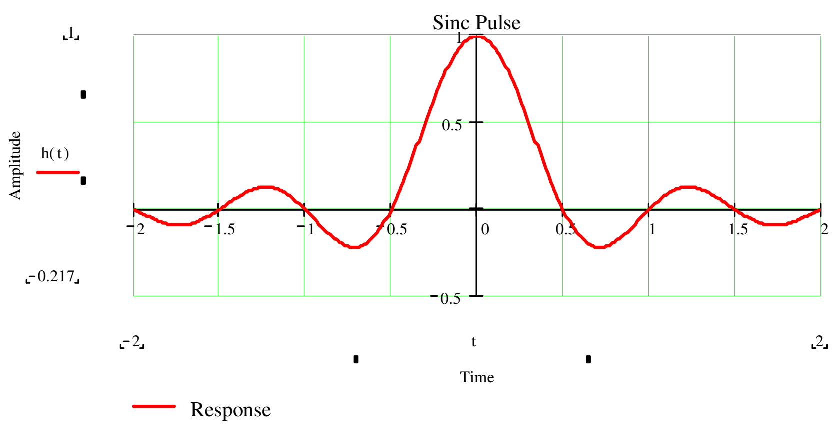
$$\mathrm{Re}\{f(t)\} = \mathrm{Re}\left\{-\frac{j}{2\pi t}\cos(\omega_o t) + \frac{1}{2\pi t}\sin(\omega_o t) + \frac{j}{2\pi t}\right\}$$
$$= \frac{1}{2\pi t}\sin(\omega_o t)$$

If we recognize $\omega_o = 2\pi f_o$, and normalize the frequency to 1, we obtain:

$$f(t) = \frac{1}{2\pi t}\sin(2\pi t) = \mathrm{sinc}(2\pi t)$$



MathCAD Sinc Pulse Simulation



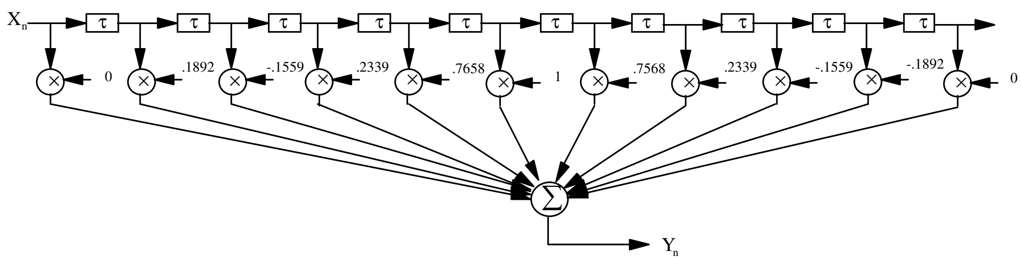
Since this function is normalized, when time = 1: $t = \frac{1}{f_o}$

To exactly reproduce this response with a FIR filter, an infinite number of coefficients and delay elements would be needed since the function stretches to infinity. Using an infinite number of delays also implies that the output signal would never emerge! So compromises must be made.

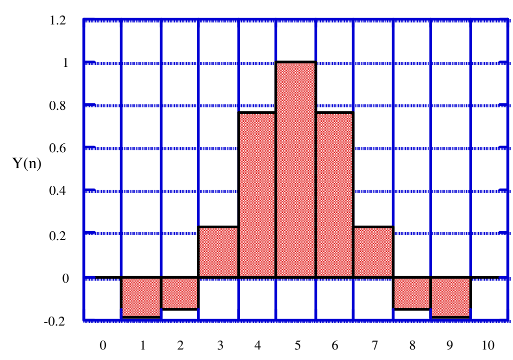
For the sake of this example, we'll arbitrarily choose 10 delay elements in the interval -1 < *t* < 1, then 11 values of the function must be determined [5 on either side of the center]. The delay element is therefore $\tau = \frac{2}{10} = .2$ seconds. The values of *f(t)* become the coefficients for the FIR filter.

t	$f(t)=\dfrac{\sin(2\pi t)}{2\pi t}$
-1.0	0.0
-0.8	- 0.1892
-0.6	- 0.1559
-0.4	0.2339
-0.2	0.7568
0	1.0
0.2	0.7568
0.4	0.2339
0.6	- 0.1559
0.8	- 0.1892
1.0	0.0

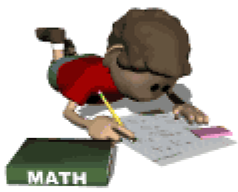
This can be implemented as:



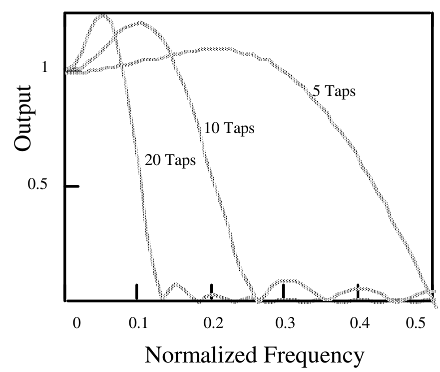
Note that this is a particularly poor implementation, since both the first and last coefficients are exactly zero and could therefore be dispensed with. The impulse response for this circuit is:



This is a crude approximation of the desired response. Nevertheless, since the impulse response is the somewhat the same, the frequency response is similar:



MathCAD FIR Filter Response

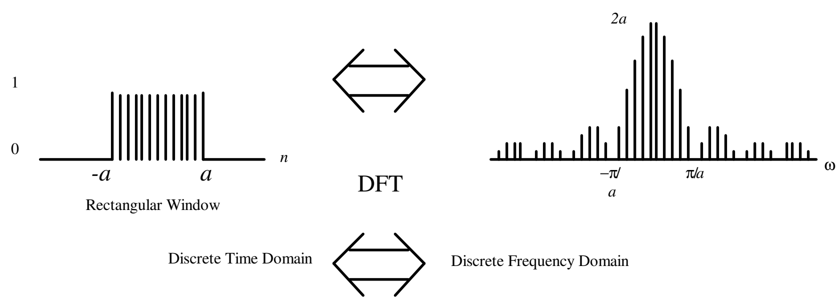


In order to improve the spectral response of this type of filter, windowing techniques are often employed.

10.1.5 Windowing

Applications Notes\ADC DAC\Windowed Sampling.pdf

A time domain window is the period during which a given signal is being processed. Since signals can only be observed for finite time intervals, the frequency is distorted by the sinc function associated with the rectangular window.



Applying a time domain window function involves performing a time domain multiplication of the filter function and window. In the frequency domain, this is manifest as a convolution. The essential steps in using windows to design a FIR filter are:

- Specify the desired filter response in the frequency domain
- Take the inverse Fourier transform to get the time domain equivalent
- Apply the window function to obtain the FIR coefficients
- Apply the Fourier transform to convert back to the frequency domain

- Check to see if the result is good enough, if not, try again

Choosing the best window function is somewhat of an art, and performing the inverse Fourier transforms is not easy. For these reasons, CAD design tools have been developed to help the filter designer.

10.1.5.1 CAD Techniques

The Remez exchange algorithm is often used to design filters. The criteria for determining FIR filter coefficients is based on minimizing the maximum errors in the specified pass and stop bands. Several iterations are performed to find the best fit polynomial, using standard Lagrange interpolation techniques.

The key parameters which the designer must provide for such programs includes:

f_p - passband cutoff frequency

f_s - stopband cutoff frequency

Δf - transition width

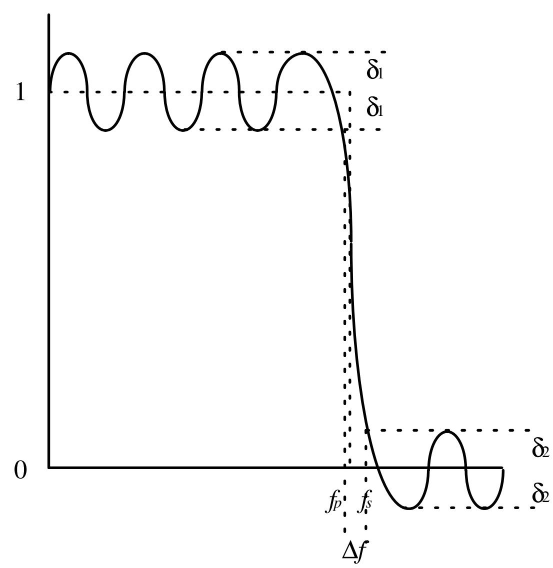
δ_1 - passband ripple

δ_2 - stopband ripple

$20 \log(1 + \delta_1)$ - passband ripple in dB

$20 \log(1 + \delta_2)$ - stopband ripple in dB

δ_1/δ_2 - ripple ratio



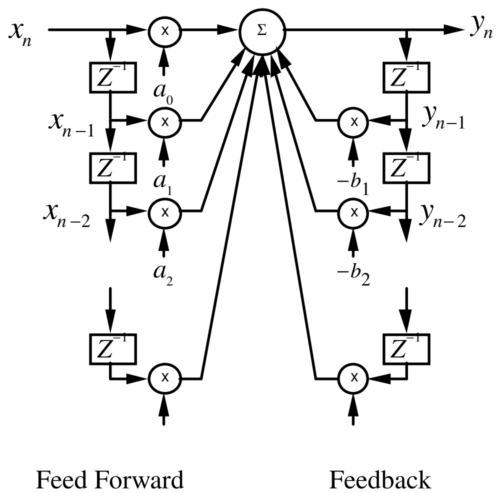
The user can often specify the number of taps or delay elements in the filter, or let the program determine a suitable value.

10.1.6 IIR Filters

The general expression for an IIR filter is given by:

$$y(k) = \underbrace{\sum_{j=0}^{m} a_j x(k-j)}_{\text{Feed Forward}} - \underbrace{\sum_{j=1}^{n} b_j y(k-j)}_{\text{Feedback}}$$

This function can be directly implemented in the following form:



The filter transfer function in the Z domain is given by:

$$H(z) = \frac{Y(z)}{X(z)} = \frac{\sum_{j=0}^{m} a_j z^{-j}}{1 + \sum_{j=1}^{n} b_j z^{-j}} = \frac{a_0 + a_1 z^{-1} + a_2 z^{-2} + \cdots a_m z^{-m}}{1 + b_1 z^{-1} + b_2 z^{-2} + \cdots b_n z^{-n}}$$

This filter has an infinite number of non-zero outputs when stimulated by an impulse, and is known as a recursive filter. The output value is determined by the present and past values of the input and the past values of the output. Some of its characteristics include:

- Potential instability
- May accumulate round-off noise
- Non-linear phase characteristics
- Feedback
- Greater efficiency than FIR filters

IIR filters can be designed by transformation or by CAD. Since the mathematics for all but the simplest filters becomes quite complex, most designs are done by CAD techniques. Never the less, it is beneficial to gain some insight in the manual methods for simple filters.

Invariably one starts with an S domain transfer function, and maps it to the Z domain. There are two popular mapping techniques currently in use:

- Impulse invariant transformation - the impulse response is determined and duplicated.
- Bilinear transformation - the digital filter duplicates the differential equation characterizing the analog filter.

10.1.6.1 Design by Transformation

The transformation method, requires the following steps:

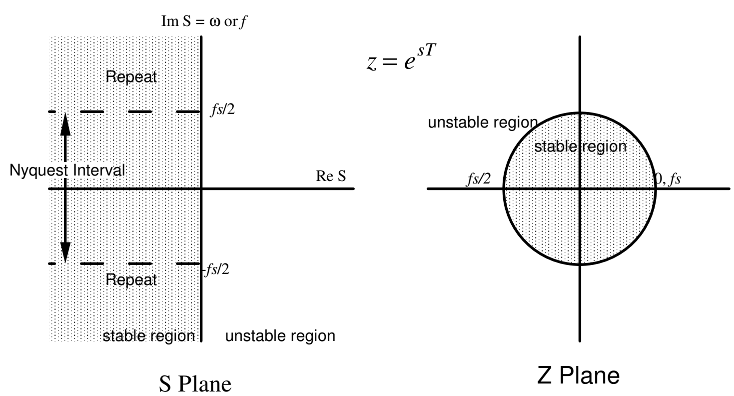
- Design an analog filter
- Derive the Laplace Transform
- Convert to the Z domain
- Apply a Z transformation [i.e. a Bilinear Transformation]
- Solve for the digital coefficients

This approach is good only for very simple filters, and introduces a measure of distortion. Two popular transformations are used to map from the S domain to the Z domain:

- Impulse invariant transform
- Bilinear transform

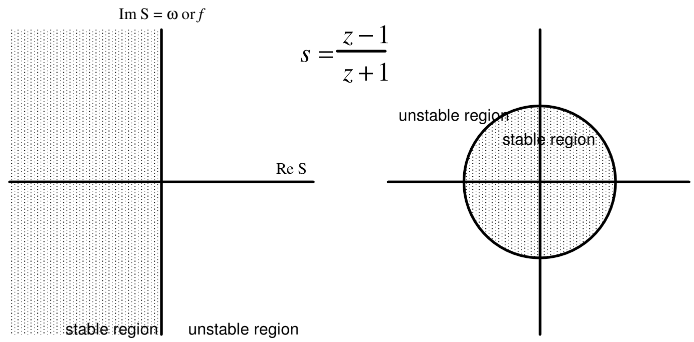
The term mapping is used to describe the process of transforming from one domain to another. This is quite appropriate if we consider the graphical representation of the two planes.

10.1.6.2 Impulse Invariant Mapping



The left hand side of the S plane maps within the unit circle on the Z plane. A distance of $\pm f_s/2$ form the horizontal axis in the S plane, wraps once around the circle in the Z plane.

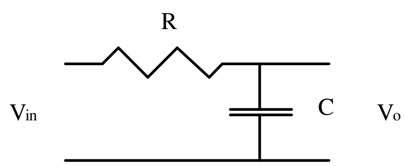
10.1.6.3 Bilinear Mapping



S Plane Z Plane

The bilinear transform maps the entire left side of the S plane into the unit circle on the Z plane. This is a better way to implement LPFs in the digital domain. Analog filters do not degenerate from aliasing and neither do bilinear mapped LPFs since the mapping is not cyclical.

An RC low pass network is a very simple circuit, and it is quite easy to design a digital equivalent.



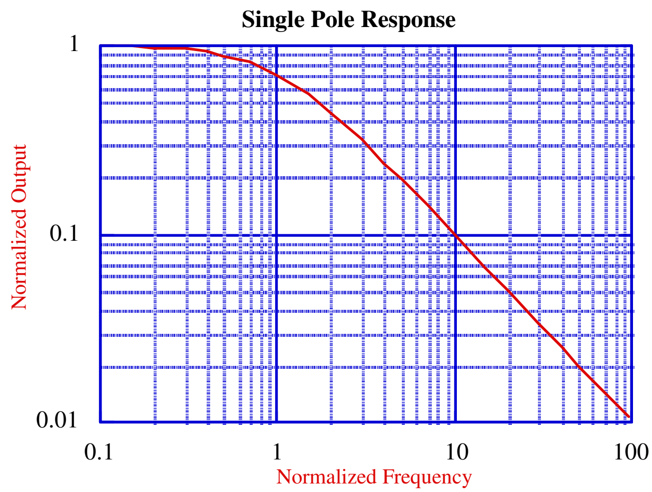
The transfer function for this circuit can be written as:

$$H(\omega) = \frac{V_o}{V_{in}} = \frac{1}{1 + RC\omega j}$$

The frequency response can be normalized in terms of the resonant or corner frequency. This allows us to compare the response of any simple network, regardless of the actual component values.

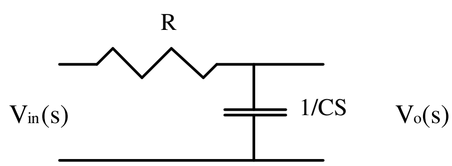
$$H(\omega)_{\text{normalized}} = \frac{1}{1 + \frac{\omega}{\omega_o} j} = \frac{1}{1 + \frac{n\omega_o}{\omega_o} j} = \frac{1}{1 + nj}$$

From this we obtain the following response:



10.1.7 Laplace Transform

The Laplace transform allows complex calculus expressions to be reduced to algebraic expressions for easier manipulation. The circuit can be transformed by means of the Laplacian operator, as follows:



The output of this circuit in the S domain is determined by applying the voltage divider rule:

$$V_{out}(s) = \frac{\frac{1}{Cs}}{R + \frac{1}{Cs}} V_{in}(s) = \frac{1}{RCs+1} V_{in}(s)$$

The *S* domain transfer function is given by:

$$H(s) = \frac{V_{out}(s)}{V_{in}(s)} = \frac{1}{RCs+1}$$

This expression can be converted to the time domain by taking the Inverse Laplace Transform. From the tables we notice that:

$$\frac{1}{s-a} \Leftrightarrow e^{at}$$

Therefore:

$$h(t) = \mathrm{L}^{-1}\left[H(s)\right] = \mathrm{L}^{-1}\left[\frac{1}{RCs+1}\right] = \mathrm{L}^{-1}\left[\frac{1}{RC}\,\frac{1}{s+\frac{1}{RC}}\right]$$

$$= \frac{1}{RC}\,e^{-t/RC}$$

10.1.8 Impulse Response

The impulse response is found by applying a delta [δ] function to the input:

$$V_{in}(t) = \delta(t)$$

The Laplace transform of the δ pulse is 1. Therefore, in the *S* domain, the input corresponds to:

$$V_{in}(s) = 1$$

and the output is the same as the transfer function:

$$H(s) = \frac{V_{out}(s)}{1}$$

As a result, the time domain response of a transfer function occurs when a delta pulse excites a circuit.

The output when the impulse strikes at $t = 0$, is the inverse of the *RC* time constant and is generally quite large. In order to compare the response of various circuits, this output excursion is usually normalized to 1.

$$V_{out}(t)_{impulse\ normalized} = e^{-t/RC}$$

The *Z* domain is the digital equivalent of the time domain. Whereas time domain is continuous, the *Z* domain is discrete. From the *Z* transform tables we note that:

$$\frac{1}{s+a} \Leftrightarrow \frac{z}{z-e^{-aT}}$$

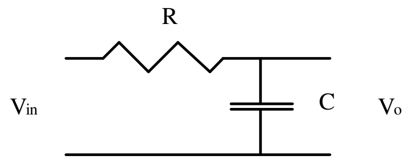
As a result, the impulse response in the *S* domain can be translated into the *Z* domain:

$$V_{out}(z)_{impulse} = \frac{1}{RC}\left(\frac{z}{z-e^{-T/RC}}\right) = \frac{1}{RC}\left(\frac{1}{1-e^{-T/RC}z^{-1}}\right)$$

The output magnitude is dependent upon the ratio $1/RC$ just as it was in the time domain. Consequently, the output function is normalized. This is generally done by evaluating the transfer function when $z = 0$ and then dividing the un-normalized transfer function by the result.

$$H(z)_{normalized} = \frac{1}{1 - e^{-T/RC} z^{-1}}$$

For the purposes of explanation, we shall assume some arbitrary component values.



Let's use as an example:

Analog:	R = 10 KΩ	C = .01 µfd
Digital:	100 K samples per Sec	Sampling time T = 10 µSec

$$\omega_o = \frac{1}{RC} = 10000 \quad \text{radians per second}$$

$$f_o = \frac{1}{2\pi RC} = 1.591549 \quad \text{KHz}$$

The transfer function for this circuit can be written as:

$$H(\omega) = \frac{V_o}{V_{in}} = \frac{1}{1 + RC\omega j}$$

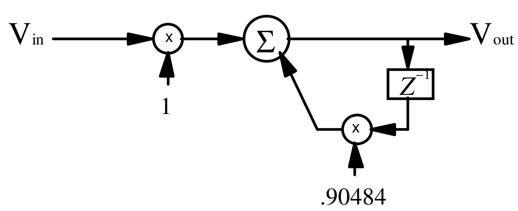
The normalized analog frequency response allows us to compare the response with a normalized digital filter.

$$H(\omega)_{\text{normalized}} = \frac{1}{1 + nj}$$

Substituting the above values for *R, C*, and *T* into the *Z* domain expression, we obtain:

$$H(z)_{normalized} = \frac{1}{1 - e^{-T/RC} z^{-1}} = \frac{1}{1 - .90484 z^{-1}}$$

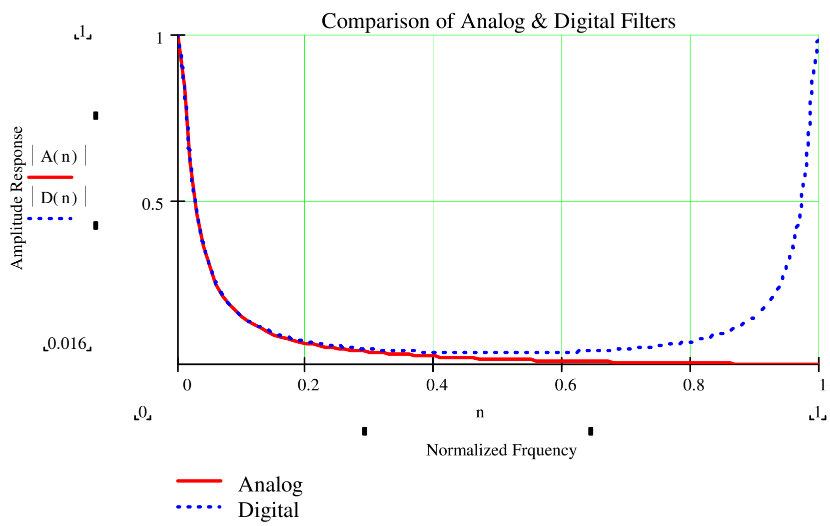
This can be implemented as:



The term Z^{-1} corresponds to a unit delay. In this example, the unit delay will be the sampling period of 10 µSec. The output from the digital filter can therefor be calculated by means of the following equation:

$$V_o[n] = V_{in}[n] + .90484 V_o[n-1]$$

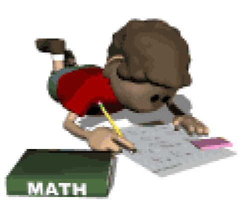
The resulting normalized frequency response is:



It would appear that the analog and digital responses are identical however, they are not.

A phenomenon known as wrap around occurs in digital filters when the input frequency approaches the sampling rate. In this particular example, the sampling rate was 100 K samples per second or 628 K radians per second.

A single pole filter using the bilinear transform



MathCAD Bilinear Transform

The bilinear transformation uses the following substitution:

$$s = \frac{2}{T}\left(\frac{z-1}{z+1}\right)$$

Recall that the circuit transfer function for this example is:

$$H(s) = \frac{V_{out}(s)}{V_{in}(s)} = \frac{1}{RCs+1}$$

Taking the bilinear transformation, we obtain:

$$H(z) = \frac{1}{\frac{2RC}{T}\left(\frac{z-1}{z+1}\right)+1} = \frac{Tz+T}{(T+2RC)z+T-2RC}$$

$$= \frac{T+Tz^{-1}}{(T+2RC)+(T-2RC)z^{-1}}$$

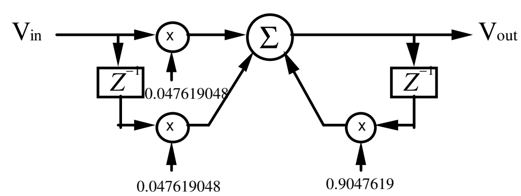
$$= \frac{\frac{T}{T+2RC}+\frac{T}{T+2RC}z^{-1}}{1+\frac{T-2RC}{T+2RC}z^{-1}}$$

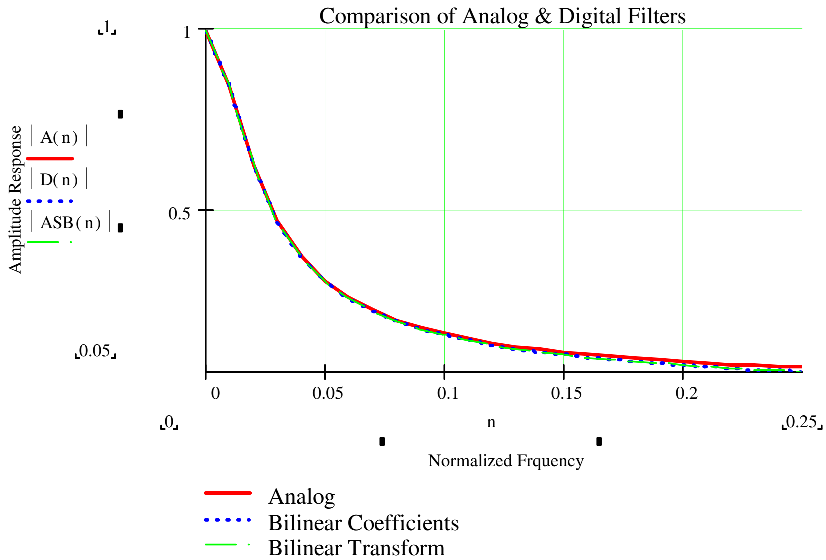
Notice that this expression does not have to be normalized since it equals one when Z = 1.

Plugging in the values for R, C, and T into this expression, we obtain:

$$H(z) = \frac{.047619048+.047619048z^{-1}}{1-.9047619z^{-1}}$$

This can be implemented as:

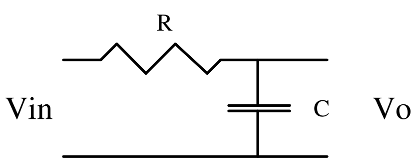




Another way to arrive at the same result is by applying Euler's approximation to rewrite differential equations as difference equations.

A single pole filter by using Euler's Approximation

This method requires transforming differential equations into difference equations. Difference equations are an algebraic approximation of differential equations.



The current in the capacitor is given by:

$$i = C\frac{dV_o}{dt}$$

Writing Kirchoff's voltage law around the loop we obtain:

$$V_{in} = Ri + V_o = RC\frac{dV_o}{dt} + V_o$$

The derivative can be evaluated by using Euler's differential approximation:

$$\frac{dV}{dt} = \frac{V[n] - V[n-1]}{T}$$

$$V_{in} = Ri + V_o = RC\frac{dV_o}{dt} + V_o$$

$$V_{in}[n] = RC\left(\frac{V_o[n] - V_o[n-1]}{T}\right) + V_o[n]$$

$$= \frac{RC}{T}V_o[n] - \frac{RC}{T}V_o[n-1] + V_o[n]$$

$$= \left(1 + \frac{RC}{T}\right)V_o[n] - \frac{RC}{T}V_o[n-1]$$

Solving for $V_O[n]$ we obtain:

$$V_o[n] = \frac{T}{RC+T}\left(V_{in}[n] + \frac{RC}{T}V_o[n-1]\right)$$

$$= \frac{T}{RC+T}V_{in}[n] + \frac{RC}{RC+T}V_o[n-1]$$

The transfer function for this circuit can be written in the Z domain as:

$$V_o(z) = \frac{T}{RC+T}V_{in}(z) + \frac{RC}{RC+T}V_o(z)z^{-1}$$

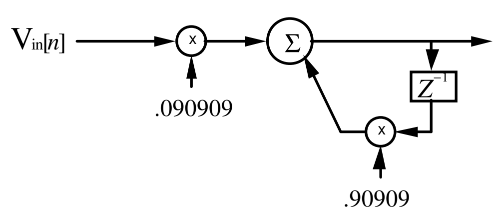
$$\frac{V_o}{V_{in}}(z) = H(z) = \frac{\frac{T}{RC+T}}{1 - \frac{RC}{RC+T}z^{-1}} = \frac{a_0}{1 - b_1 z^{-1}}$$

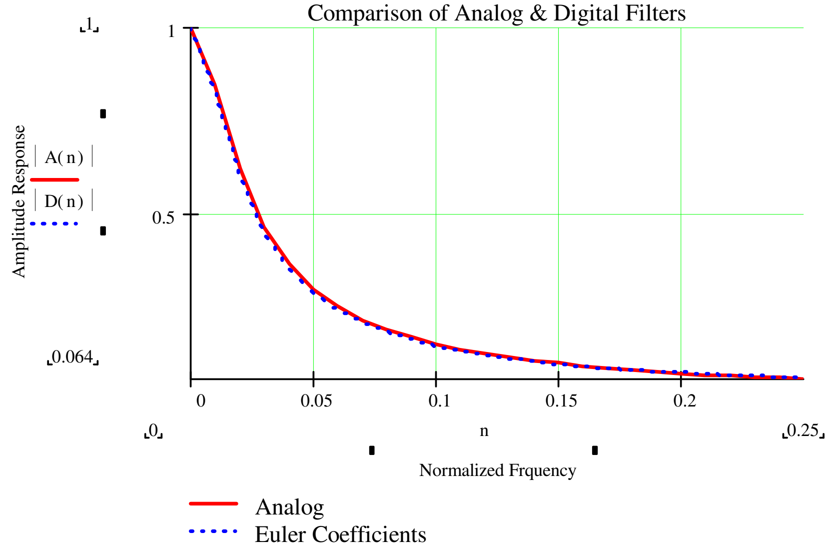
For the same values of R, C, and T, we obtain:

$$V_o[n] = .090909V_{in}[n] + .90909V_o[n-1]$$

$$H(z) = \frac{.090909}{1 - .90909Z^{-1}}$$

These values are very similar to those found in example 3. Consequently, the implementation is the same.





A single pole filter using another transform

The S domain form of the transfer function can also be converted to the *Z* domain by making the following substitution:

$$s = \frac{1 - z^{-1}}{T}$$

Recall that the z^1 term can be interpreted as a delay equal to the sampling period *T*. Therefore:

$$\frac{V_{out}(s)}{V_{in}(s)} = \frac{1}{RCs + 1}$$

$$\frac{V_{out}(z)}{V_{in}(z)} = \frac{1}{RC\frac{1 - z^{-1}}{T} + 1}$$

$$V_{out} = \frac{1}{RC\frac{1 - z^{-1}}{T} + 1} V_{in} = \frac{T}{T + RC - RCz^{-1}} V_{in}$$

$$V_{out}\left(T + RC - RCz^{-1}\right) = TV_{in}$$

Collecting all of the *Z* terms to the right hand side, we obtain:

$$V_{out}(T + RC) = TV_{in} + RCz^{-1}V_{out}$$

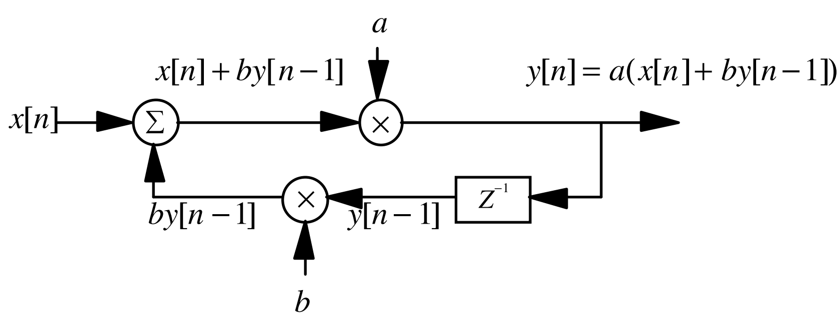
Solving for V_O, we obtain:

$$V_{out} = \frac{T}{T + RC} V_{in} + \frac{RC}{T + RC} z^{-1} V_{out}$$

This expression is identical to the one derived previously.

Alternate Configuration

There are often several ways to implement the same filter response. Another configuration, which emulates a simple RC network, is:



This circuit can easily be characterized by placing a step function at the input.

If the input is a step function starting at n = 1, then:

$$x[n] = 0 \qquad \text{for} \qquad n \le 0$$
$$x[n] = 1 \qquad \text{for} \qquad n > 0$$

Since the filter is causal [the output cannot anticipate the input], $y[0] = 0$, then:

$$n = 1 \qquad y[1] = a\left(x[1] + by[0]\right) = a$$
$$n = 2 \qquad y[2] = a\left(1 + by[1]\right) = a(1 + ab)$$
$$n = 3 \qquad y[3] = a\left(1 + by[2]\right) = a\left(1 + b\left(a(1 + ab)\right)\right) = a\left(1 + ab + (ab)^2\right)$$
$$n = n \qquad y[n] = 1\left(1 + ab + (ab)^2 + (ab)^3 + \cdots (ab)^{n-1}\right) = a\frac{1-(ab)^n}{1-ab}$$

If ab<1, then this filter arrangement has a response similar to a simple RC network:

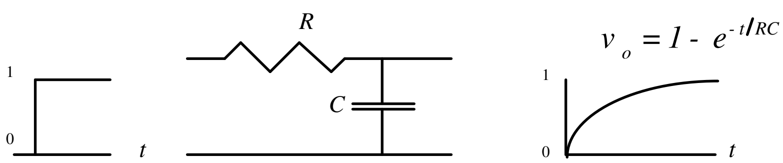
$$y[n] = \frac{a}{1-ab}\left(1 - (ab)^n\right) \quad \text{is similar to} \quad v_o = v_i\left(1 - e^{-t/RC}\right)$$

or in terms of a sampled network:

$$y[n] = \frac{a}{1-ab}\left(1 - (ab)^n\right) \quad \text{is similar to} \quad v[n] = 1 - e^{-nT/RC}$$

$$\text{when} \quad \frac{a}{1-ab} = 1 \quad \text{and} \quad (ab)^n = e^{-nT/RC}$$

Applying a step function to a simple RC circuit, we obtain:



Input

Output

For the same values of R, C, and T, we obtain the coefficients a & b:

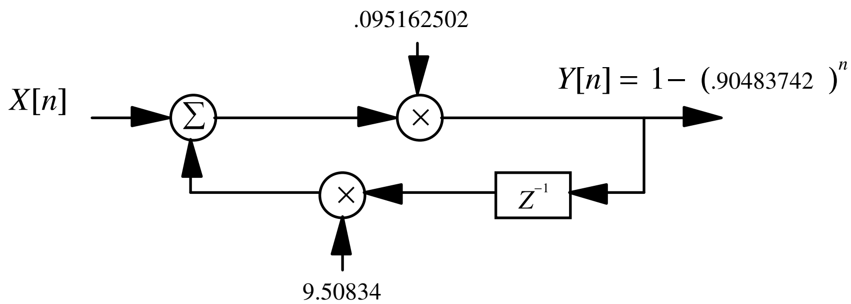
Since: $(ab)^n = e^{-nT/RC}$

Therefor $ab = e^{-T/RC} = .90483742$

And since $\dfrac{a}{1-ab} = 1$, therefor $\dfrac{a}{1-.90483742} = 1$ and

Consequently $a = .095162502$ and $b = 9.50834$

These are the coefficients needed to make the filter work.



Creating digital filters by trial and error is quite unsatisfactory, and therefore more rigorous methods have been developed.

10.1.9 Higher Order Filters

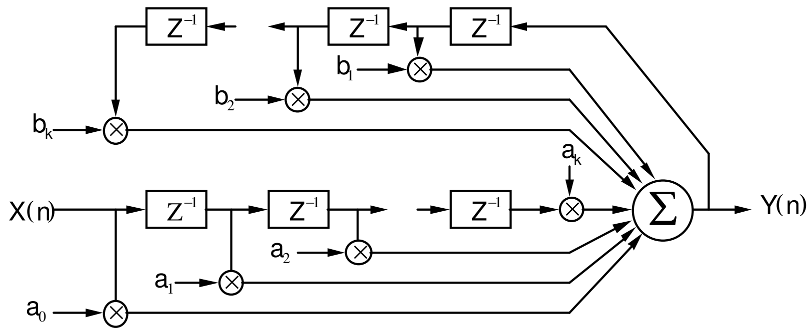
The discrete transfer function form:

$$\frac{Y(z)}{X(z)} = H(z) = \frac{\sum_{j=0}^{m} a_j z^{-j}}{1 + \sum_{j=1}^{n} b_j z^{-j}} = \frac{a_0 + a_1 z^{-1} + a_2 z^{-2} + \cdots a_m z^{-m}}{1 + b_1 z^{-1} + b_2 z^{-2} + \cdots b_n z^{-n}}$$

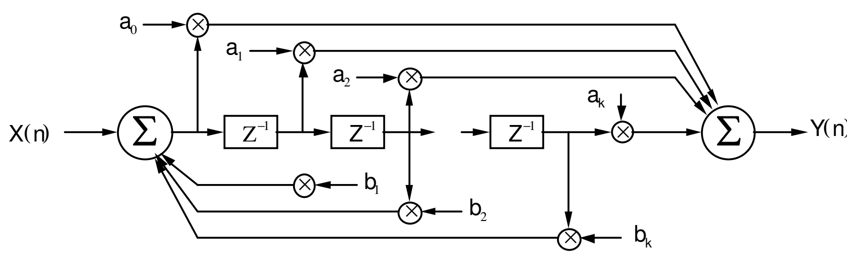
can be used to implement digital filters directly by inspection.

_____ Notes _____

10.1.9.1 Direct form 1 implementation:



10.1.9.2 Direct form 2 implementation:

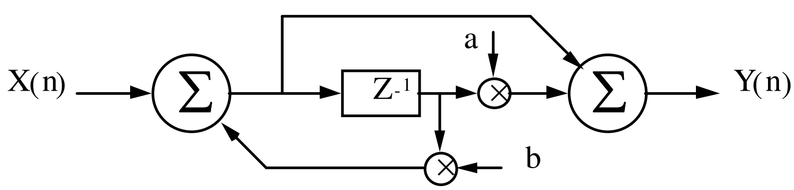


As the transfer function order increases, the coefficient sensitivity also increases. To overcome this phenomenon, complex transfer functions are often decomposed into series [cascade] or parallel, first and second order segments.

A first order section can be written:

$$H(z) = \frac{1 + az^{-1}}{1 + bz^{-1}}$$

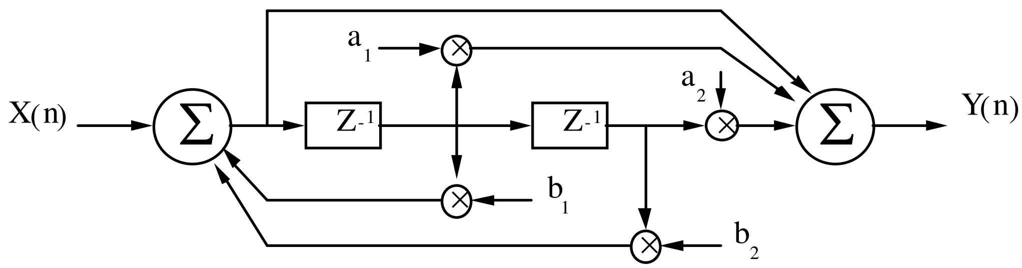
and implemented as:



A second order section can be written:

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

and implemented as:

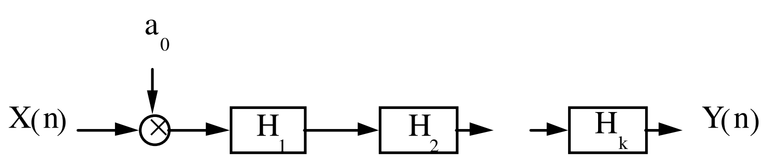


10.1.9.3 Cascade Realizations

It is not practical to design high order filters as a single unit. It is more practical to decompose a complex transfer function into second order sums and products. A complex transfer function can often be decomposed into products:

$$H(z) = \frac{Y(z)}{X(z)} = a_0 H_1(z) H_2(z) \ldots H_k(z)$$

and implemented as:



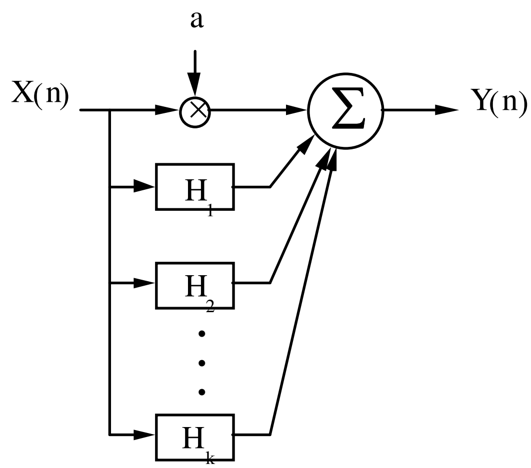
Each block [*H*] represents a second order component of the overall function.

10.1.9.4 Parallel Realizations

In some cases, it is more practical to decompose the complete function into a series of sums.

$$H(z) = \frac{Y(z)}{X(z)} = a + H_1(z) + H_2(z) + \ldots + H_k(z)$$

and implemented as:

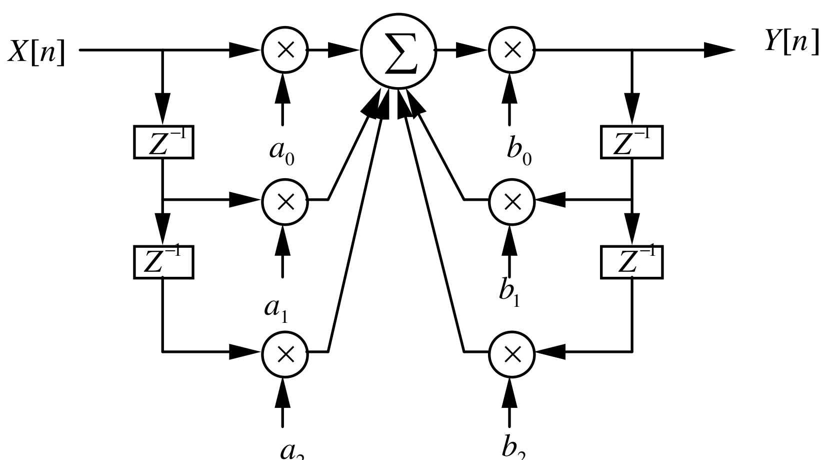


One simple application for this type of network is a graphic equalizer. Each functional filter block can be a digital band pass filter designed to respond to a specific frequency band.

10.1.10 Design by Synthesis:

The biquad is the preferred building block used to design digital filters by computer.

- A CAD method
- Coefficients and structure are optimized for a specific performance
- Biquads are the preferred structure due to mathematical considerations [but minimizes the number of biquads]



Biquad

The transfer function for the biquad if $b_0 = 1$ is given by:

$$\frac{Y[z]}{X[z]} = \frac{a_0 + a_1 z^{-1} + a_2 z^{-2}}{1 - b_1 z^{-1} - b_2 z^{-2}}$$

To evaluate the frequency response, let $z = e^{j\omega T}$.

10.1.10.1 Butterworth LPF Design

Digital Filters can be designed from tables of various coefficients. One of the most common of these is based on the Butterworth function. The Butterworth response is also called maximally flat. The normalized amplitude response of an *n*th order filter of this type is given by:

$$\left|H_n(j\omega)\right| = \frac{1}{\sqrt{1+\omega^{2n}}} \qquad n = 1, 2, 3 \ldots$$

Coefficients of Butterworth Polynomials

n	a_1	a_2	a_3	a_4	a_5	a_6	a_7
2	1.4142						
3	2.0	2.0					
4	2.6131	3.4142	2.6131				
5	3.2361	5.2361	5.2361	3.2361			
6	3.8637	7.4641	9.1416	7.4641	3.8637		
7	4.4940	1.00978	14.5918	14.5918	1.00978	4.4940	
8	5.1528	13.1371	21.8462	25.6884	21.8462	13.1371	5.1528

Factors of Butterworth Polynomials

n	Factored Polynomial
1	$s+1$
2	$s^2+1.4142s+1$
3	$(s+1)(s^2+s+1)$
4	$(s^2+0.7654s+1)(s^2+1.8478s+1)$
5	$(s+1)(s^2+0.6180s+1)(s^2+1.6180s+1)$
6	$(s^2+0.5176s+1)(s^2+1.4142s+1)(s^2+1.9319s+1)$
7	$(s+1)(s^2+0.4450s+1)(s^2+1.2470s+1)(s^2+1.8019s+1)$
8	$(s^2+0.3902s+1)(s^2+1.1111s+1)(s^2+1.6639s+1)(s^2+1.9616s+1)$
9	$(s+1)(s^2+0.3473s+1)(s^2+s+1)(s^2+1.5321s+1)(s^2+1.8794s+1)$

10.1.10.2 3rd Order Butterworth LPF

Design a Butterworth LPF where the sampling frequency is 15 times the cutoff frequency[1] .

Since: $f_s = 15 f_c$ or $\omega_s = 15\omega_c$

but $\omega_s = \dfrac{2\pi}{T}$ where *T* is the sampling rate.

1 See MathCad file: 3rd Order Butterworth

If the cutoff frequency is normalized to unity: $\omega_c = 1$

Then: $T = \dfrac{2\pi}{15} = 0.41888$

From the table of coefficients, the LPF transfer function can be written as:

$$H_n(s) = \frac{1}{1 + 2s + 2s^2 + s^3}$$

Or from the factored polynomial table it can be written as:

$$H_n(s) = \frac{1}{(s+1)\left(s^2 + s + 1\right)}$$

In order to transform this expression into the *Z* domain, it is necessary to write this function in the form:

$$H_n(s) = \frac{1}{(s+a)(s+b)(s+c)}$$

and then break it up by the process of partial fractions:

$$H_n(s) = \frac{1}{(s+a)(s+b)(s+c)} = \frac{A}{s+a} + \frac{B}{s+b} + \frac{C}{s+c}$$

It is obvious that: $a = 1$ and that the b and c terms can be found by using the quadratic formula:

$$b = \frac{1-\sqrt{3}j}{2} \qquad \text{and} \qquad c = \frac{1+\sqrt{3}j}{2}$$

Therefore:

$$H_n(s) = \frac{1}{(s+1)\left(s + \dfrac{1-\sqrt{3}j}{2}\right)\left(s + \dfrac{1+\sqrt{3}j}{2}\right)}$$

Finding the coefficients *A, B,* and *C* is relatively straightforward. The process involves making substitutions for *s* such that only one coefficient remains in the equation.

$$\frac{A}{s+a} + \frac{B}{s+b} + \frac{C}{s+c} = \frac{1}{(s+a)(s+b)(s+c)}$$
$$A(s+b)(s+c) + B(s+a)(s+c) + C(s+a)(s+b) = 1$$

To find *A*, let $s = -a$:

$$A(-a+b)(-a+c)=1$$

$$A=\frac{1}{(-a+b)(-a+c)}=\frac{1}{\left(-1+\frac{1-\sqrt{3}j}{2}\right)\left(-1+\frac{1-\sqrt{3}j}{2}\right)}=1$$

To find *B*, let *s* = *-b:*

$$B(-b+a)(-b+c)=1$$

$$B=\frac{1}{(-b+a)(-b+c)}=\frac{1}{\left(-\frac{1-\sqrt{3}j}{2}+1\right)\left(-\frac{1-\sqrt{3}j}{2}+\frac{1+\sqrt{3}j}{2}\right)}$$

$$=\frac{2}{-3+\sqrt{3}j}$$

To find *C*, let *s* = *-c:*

$$C(-c+a)(-c+b)=1$$

$$C=\frac{1}{(-c+a)(-c+b)}=\frac{1}{\left(-\frac{1+\sqrt{3}j}{2}+1\right)\left(-\frac{1+\sqrt{3}j}{2}+\frac{1-\sqrt{3}j}{2}\right)}$$

$$=\frac{2}{-3-\sqrt{3}j}$$

Putting it all together, we obtain:

$$H_n(s)=\frac{1}{s+1}+\frac{\frac{2}{-3+\sqrt{3}j}}{s+\frac{1-\sqrt{3}j}{2}}+\frac{\frac{2}{-3-\sqrt{3}j}}{s+\frac{1+\sqrt{3}j}{2}}$$

This expression can now readily be converted to the time domain to give the impulse response, or to the *Z* domain to create a digital filter.

The impulse response can be determined by applying the following Laplace transform:

$$\frac{1}{s+a}\Leftrightarrow e^{-at}$$

Therefore:

$$\frac{A}{s+a}+\frac{B}{s+b}+\frac{C}{s+c}\Leftrightarrow Ae^{-at}+Be^{-bt}+Ce^{-ct}$$

$$h(t)=e^{-t}+\frac{2}{-3+\sqrt{3}j}e^{-\frac{1-\sqrt{3}j}{2}t}+\frac{2}{-3-\sqrt{3}j}e^{-\frac{1+\sqrt{3}j}{2}t}$$

The S domain function can also be transformed into the Z domain by the following transform:

$$\frac{1}{s+a} \Leftrightarrow \frac{z}{z-e^{aT}}$$

Therefore:

$$\frac{A}{s+a}+\frac{B}{s+b}+\frac{C}{s+c} \Leftrightarrow A\frac{z}{z-e^{-aT}}+B\frac{z}{z-e^{-bT}}+C\frac{z}{z-e^{-cT}}$$

$$H(z)=\frac{z}{z-e^{-T}}+\frac{2}{-3+\sqrt{3}j}\,\frac{z}{z-e^{-\frac{1-\sqrt{3}j}{2}T}}+\frac{2}{-3-\sqrt{3}j}\,\frac{z}{z-e^{-\frac{1+\sqrt{3}j}{2}T}}$$

At this point, the partial fraction expansion must now be converted back to a single fraction in order to be readily implemented as a digital filter. This is a very straightforward but tedious process.

$$H(z)=A\frac{z}{z-e^{-aT}}+B\frac{z}{z-e^{-bT}}+C\frac{z}{z-e^{-cT}}$$

$$=\frac{(A+B+C)z^3-\left[A\left(e^{-bT}+e^{-cT}\right)+B\left(e^{-aT}+e^{-cT}\right)+C\left(e^{-aT}+e^{-bT}\right)\right]z^2+\left(Ae^{-(b+c)T}+Be^{-(a+c)T}+Ce^{-(a+b)}\right.}{z^3-\left(e^{-aT}+e^{-bT}+e^{-cT}\right)z^2+\left(e^{-(a+b)T}+e^{-(b+c)T}+e^{-(a+c)T}\right)z+e^{-(a+b+c)T}}$$

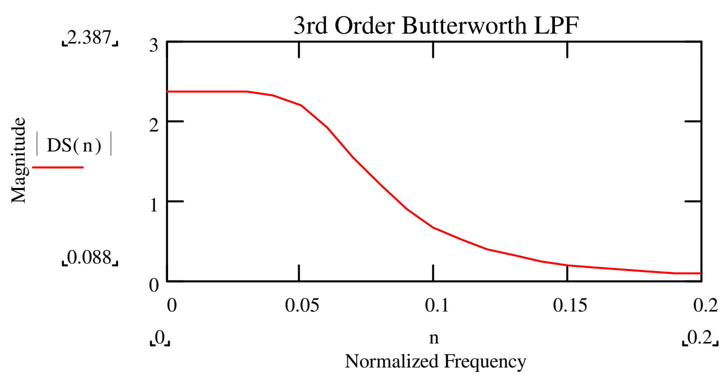
Substituting the calculated values of *a, b, c, A, B, C,* and *T* into the above expression yields:

$$H(z)=\frac{0.065689095z^2+0.0497155652z}{z^3-2.1742980352z^2+1.6553222385z-0.4326794865}$$

Dividing numerator and denominator by the highest order Z exponent [z^3] yields the standard form:

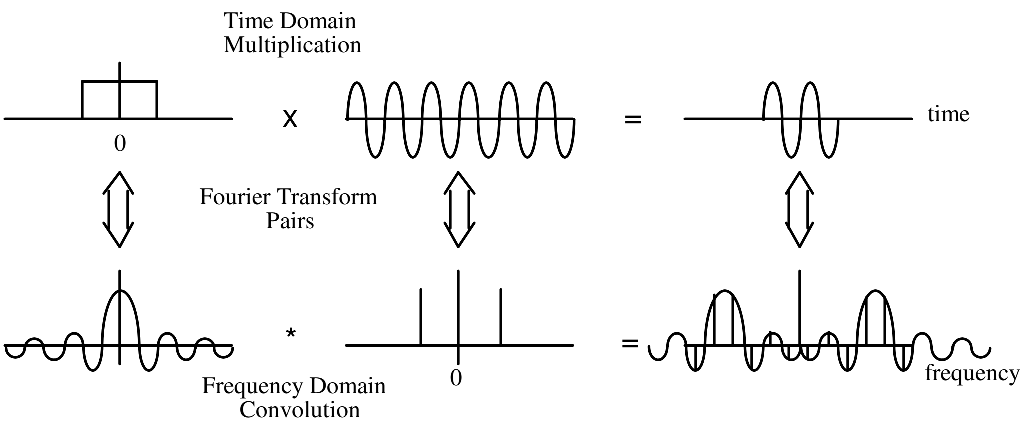
$$H(z)=\frac{0.065689095z^{-1}+0.0497155652z^{-2}}{1-2.1742980352z^{-1}+1.6553222385z^{-2}-0.4326794865z^{-3}}$$

The frequency response of this filter is:



10.2 Convolution

The time and frequency domains are related by the Fourier transform. There is also a correspondence between operations performed in each domain. For example, multiplication performed in one domain corresponds to convolution in the other.



From the above illustration, it is apparent that the frequency content of a gated sinewave or tone burst can be derived by two basic methods:

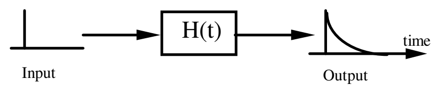
- Multiply the two time domain functions together and take the Fourier transform of the result.
- Convolve the two frequency domain functions

It depends entirely on the application as to whether it is more practical to perform the multiplication or the convolution. It is not possible to perform frequency domain multiplication directly in the digital domain, but it is possible to perform a convolution and then apply the appropriate transform to get back to the right domain.

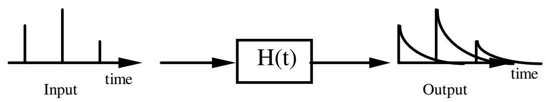
The term convolution means ‘folding back’. One way to think of the instantaneous response of a system it as a succession of individual responses, each of which is affected by what has happened before.

Since the Fourier Transform of the impulse response of a system completely characterizes the system, it is the ideal function to use as the bases for performing a convolution.

If a system responds to an impulse in some predictable fashion:



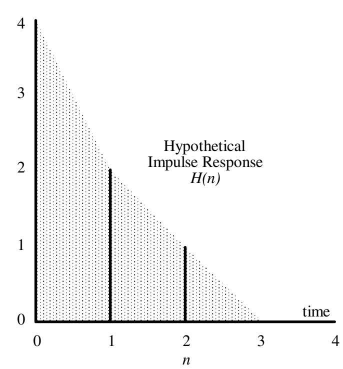
Then the response to any arbitrary input is composed of the individual responses:



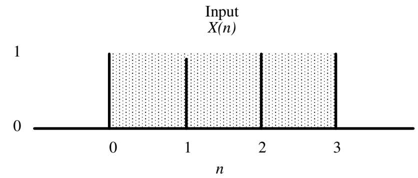
Each successive input stimulus causes an output response. As successive inputs come closer together, the output responses start to pile up. The total output response becomes the sum of each current response plus whatever response is left over from the previous stimulus. Notice that each successive output interacts with the previous response, backwards. If the input is a continuous function, the output can be determined by summing, or integrating, the product of the input signal and the time reversed impulse response.

This seems at first to be an indefensible proposition, but perhaps a worked example might illustrate. Let’s say that the impulse response in some system is given by:

A Hypothetical Impulse Response	
n	Output
0	4
1	2
2	1
3	0



If this system gets excited by a pulse:



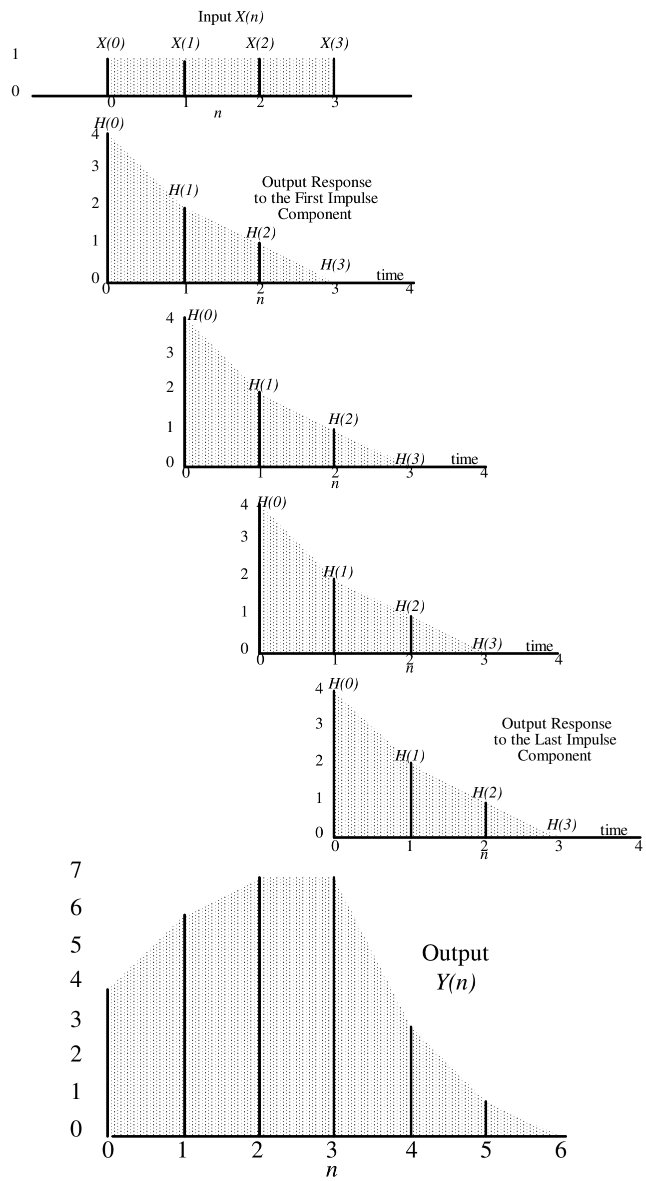
Each component will initiate this response, and the total output will be the sum of the individual responses. The product of any two signals can be determined by a sample-by-sample multiplication:

$$Y(n) = X(n)H(n)$$

The convolution of two signals is found by computing the sum products:

$$Y(n) = X(n) * H(n) = \sum_{j=-\infty}^{\infty} X(n)H(n-j)$$

This can be seen in the following illustration:



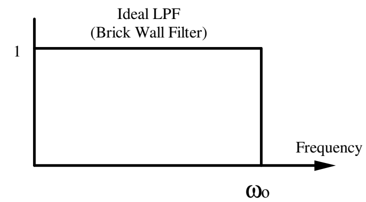
Notice how the output response is built-up:

n	$Y(n)$
0	$X(0)H(0)$
1	$X(0)H(1)+X(1)H(0)$
2	$X(0)H(2)+X(1)H(1)+X(2)H(0)$
3	$X(0)H(3)+X(1)H(2)+X(2)H(1)+X(3)H(0)$
4	$X(1)H(3)+X(2)H(2)+X(3)H(1)$
5	$X(2)H(3)+X(3)H(2)$
6	$X(3)H(3)$

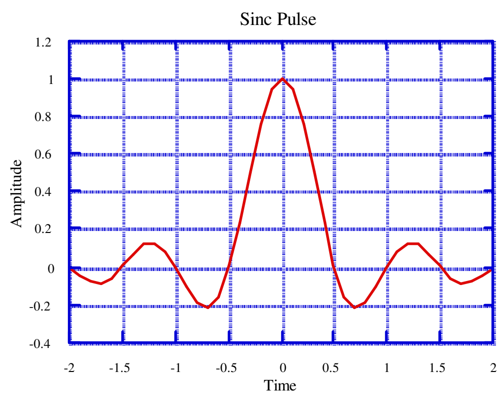
From the above table, we note that the input function $X(n)$ is incrementing in n while $H(n)$ is decrementing in n. It is for this reason, that the convolution is the integral or sum of the product of the forward input function and the reverse of the transfer function.

This means that all filters whether analog or digital, perform time domain convolutions and frequency domain products.

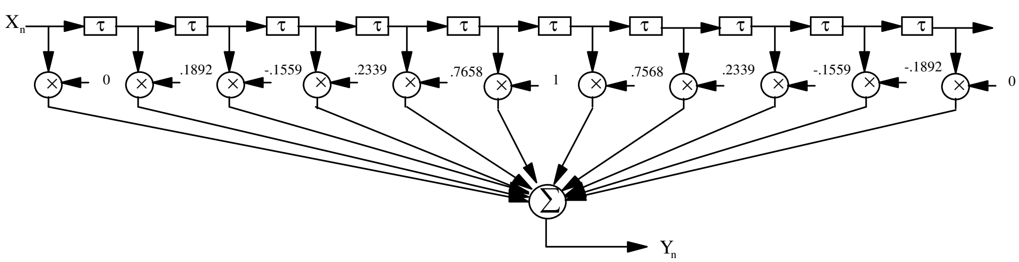
Recall that at the beginning of this discussion, a FIR filter was designed to emulate a brick wall response:



The impulse response of this system is:

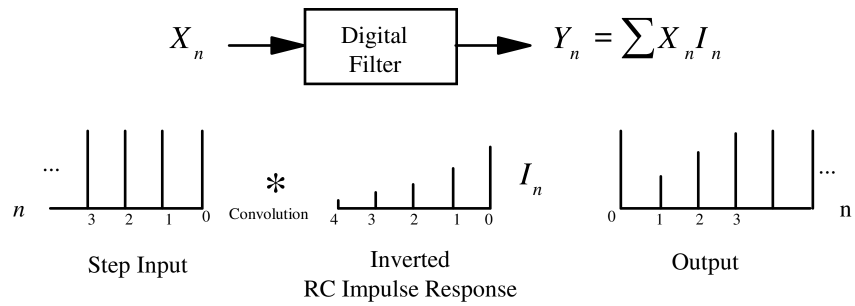


A FIR filter approximating this response is:



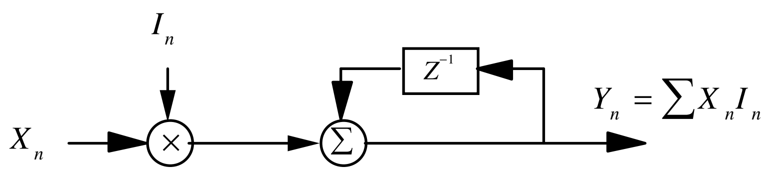
This filter is performing a convolution. In order to convert the time domain output response back to the frequency domain, it is only necessary to perform the inverse Fourier transform.

A convolution can also be done with an IIR filter by inverting the impulse response and integrating it with the input sequence. This integration process can be implemented as a summer.



$$Y_n = \sum X_n I_n$$
$$Y_0 = X_0 I_0$$
$$Y_1 = X_1 I_1 + X_0 I_0 = X_1 I_1 + Y_0$$
$$Y_2 = X_2 Y_2 + X_1 I_1 + X_0 I_0 = X_2 I_2 + Y_1$$
$$Y_n = X_n Y_n + Y_{n-1}$$

This can be implemented as:



Note that the response of this filter is identical to step function response.

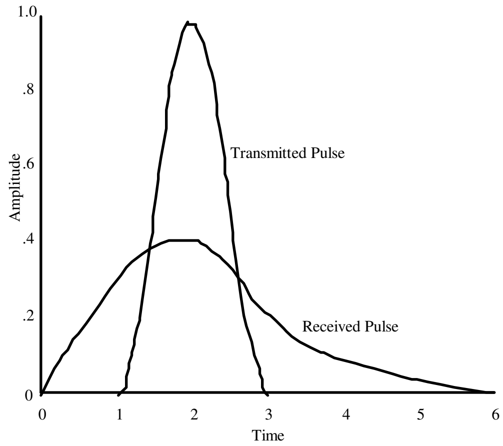
n	X_n	I_n	Y_n
0	0	0	0
1	1	0.393	0.3930
2	1	0.2383	0.6313
3	1	0.1445	0.7758
4	1	0.0876	0.8634
5	1	0.0531	0.9165
6	1	0.0322	0.8487
7	1	0.0195	0.9682
8	1	0.0118	0.9800
9	1	0.0072	0.9872
10	1	0.0044	0.9916

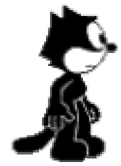
In this implementation requires a single delay element and multiplication thus reducing the throughput delay. However, what was once a constant coefficient has now changed to a sequence of values, thus requiring more memory. It should also be evident that some sort of synchronization is required in order to make any sense of the output.

10.2.1 Zero Forcing Equalizer

One of the main uses for a FIR filter is to restore a distorted received signal to something closer to its original transmitted shape.

_____ Notes _____


1.0
.8
.6
.4
.2
0
Amplitude
Transmitted Pulse
Received Pulse
0
1
2
3
4
5
6
Time



Review Questions

Quick Quiz

1. Give 4 reasons why DSP chips are gaining popularity over traditional analog devices.

   a) ____________________

   b) ____________________

   c) ____________________

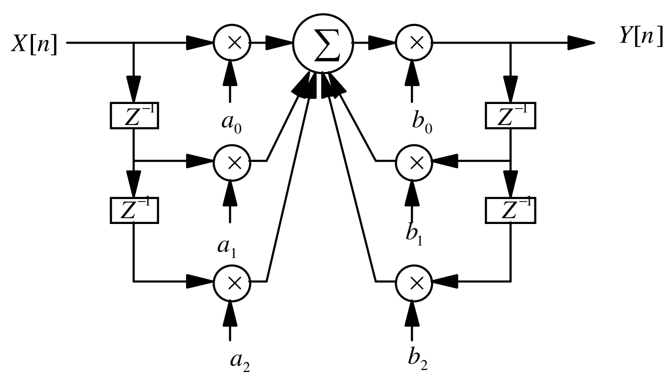
   d) ____________________

2. What two mapping techniques are most commonly used in designing low pass digital filters?

   a) ____________________

   b) ____________________

3. Identify the following circuit:



Circuit: ____________________

4. What is the fundamental difference between direct form 1 and 2 IIR filters?

   ____

   ____

   ____

5. Multiplication in the time domain corresponds to ________ in the frequency domain.

Analytical Problems

1. Design a simple recursive filter using Euler's Approximation to emulate a single pole network with R = 2.5 KΩ and C = .01μfd

2. Given that the impulse response of an RC network in the *S* domain is:

$$H(s) = \frac{1}{RCS + 1} \quad \text{where} \quad R = 1\,K\Omega \quad C = .01\,\mu\text{fd} \quad T = 10\,\mu\text{sec}$$

   Design an equivalent digital filter by applying the bilinear transform.

3. Given the sketch for the biquad, derive the following transfer function when $b_0 = 1$:

$$\frac{Y[z]}{X[z]} = \frac{a_0 + a_1 z^{-1} + a_2 z^{-2}}{1 - b_1 z^{-1} - b_2 z^{-2}}$$

Composition Questions

1. What is a recursive filter?
2. What are some of the advantages and disadvantages of a digital filter?
3. What is a FIR filter?

For Further Research 

Helgert Hermann J, *Integrated Services Digital Networks*, Addison-Wesley, New York, (1991)

Higgins Richard J, *Digital Signal Processing in VLSI*, Prentice-Hall, 1990

Jackson Leland B, *Digital Filters and Signal Processing*, Kluwer Academic Publishers, Boston, 1986

Lam Harry Y F, *Analog and Digital Filters: Design and Realization*, Prentice-Hall, 1979

Meade M L, & Dillon C R, *Signals and Systems*, Chapman & Hall, 1991

Mitra Sanjit K, & Kaiser James F, *Handbook for Digital Signal; Processing*, John Wiley & Sons, New York, 1993

Oppenheim Alan V, & Schafer Ronald W, *Discrete-Time Signal Processing*, Prentice-Hall, 1989

Poularikas Alexander D, & Seely Samuel, *Signals and Systems*, PWS Publishers, Boston, 1985

Rabinier Lawrence R, & Gold Bernard, *Theory and Application of Digital Signal Processing*, Prentice-Hall, 1975

Taylor Fred J, *Digital Filter Design Handbook*, Marcel Dekker, New York, 1983

Mixed Signal Design Seminar, Analog Devices, Norwood, MA, 1991

DSP Applications Seminar, Analog Devices, Norwood, MA, 1984

Techniques in Discrete-Time Signal Processing, Interactive Circuits and Systems LTD., March 30, 1982

http://www.zsp.com/

http://www.alacron.com/

http://www-dsp.ee.qub.ac.uk/

http://allegro.mit.edu/

http://www.intermetall.de/

http://www.asti-usa.com/index.html

http://dsp.chonnam.ac.kr/

http://tokyo.usask.ca/dsp.htm

http://www.aspi.com/