Massachusetts Institute of Technology
Dept. of Electrical Engineering and Computer Science
Fall Semester, 2006
6.082 Introduction to EECS 2

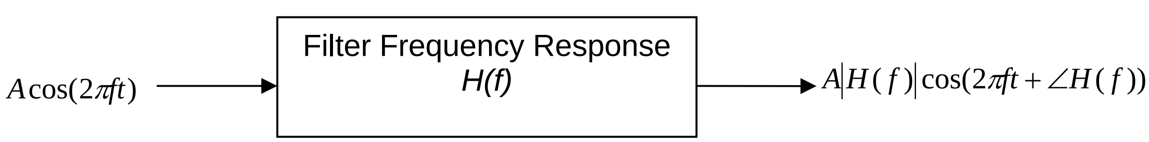
Filtering

Introduction

Filtering is a part of every signal processing application. Filtering involves processing a signal in order to *attenuate or suppress* some of its frequency components while *passing* another set of its frequency components unaffected.

The degree to which a filter passes or attenuates a particular frequency component is summarized by the filter's frequency response *H(f)*. The quantity *H(f)* is a complex number; it has a magnitude $|H(f)|$ and a phase $\angle H(f)$.

When a sinusoid with amplitude *A* and frequency *f* passes through a filter, the sinusoid emerges with a frequency *f;* an amplitude $A|H(f)|$; and is phase shifted by $\angle H(f)$. In other words, the frequency of the resulting sinusoid is unchanged, but the amplitude of the sinusoid is scaled by the magnitude of the filter frequency response $|H(f)|$, and the phase of the sinusoid is shifted by phase of the filter frequency response $\angle H(f)$. The action of a filter on a sinusoid is summarized below.



Through these lecture notes you will study different filters types; visualize the effects of filters in the time and frequency domains; and understand how filters are implemented using difference equations. Throughout these lecture notes we will be concerned primarily with characterizing the magnitude of the filter frequency response $|H(f)|$; you will learn to characterize the phase of the filter frequency response $\angle H(f)$ in future courses.

Filter Types

Low Pass Filters

Consider a signal x(t) formed by the addition of unity amplitude sinusoids with frequencies of 10 Hz, 50 Hz, and 90 Hz as illustrated in equation 1. Examine the time and frequency-domain representations of this signal shown Figures 1 and 2.

$$x(t) = \sin(2\pi * 10 * t) + \sin(2\pi * 50 * t) + \sin(2\pi * 90 * t) \qquad (1)$$

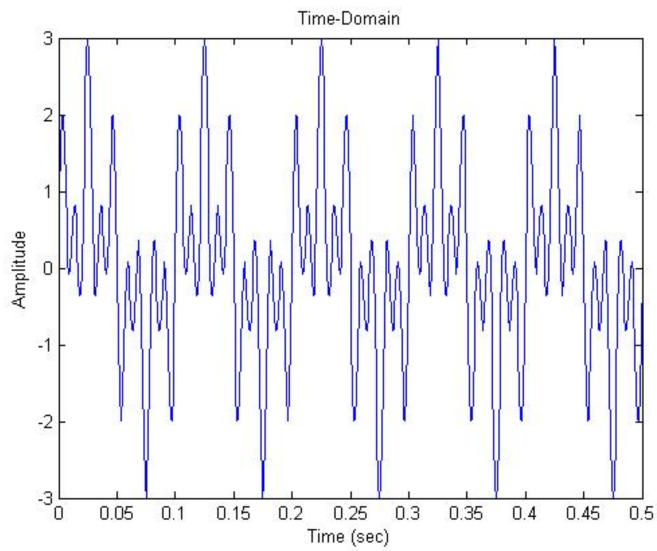


Figure 1

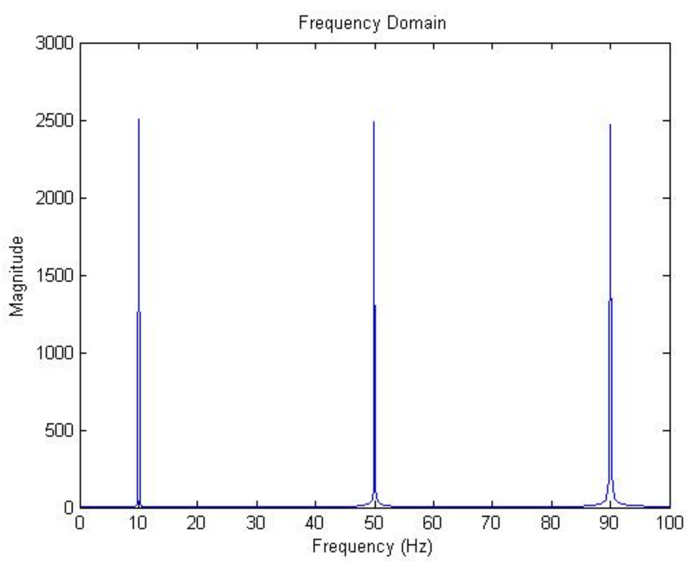


Figure 2

Now consider filtering the input x(t) so that we retain the 10 Hz component and attenuate both the 50 Hz and 90 Hz components. To do this we will use a *low pass filter*. A low pass filter passes all frequency components of a signal that are smaller than a *cutoff frequency* f_c, and attenuates all frequency components that are greater than the cutoff. In this example, we will use a low pass filter with a cutoff frequency f_c=25 Hz in order to pass the 10 Hz component (10 Hz < f_c so this frequency component passes) and attenuate both the 50 Hz and 90 Hz components (90 Hz > 50 Hz > f_c so these frequency components are attenuated).

Figure 3 illustrates *the magnitude of the frequency response* of our low pass filter. The magnitude of the frequency response of a filter is used to determine how a filter will affect the magnitude of different frequency components in a signal. More specifically, for each possible frequency component the frequency response indicates the fraction of the amplitude of a frequency component that will pass through the filter.

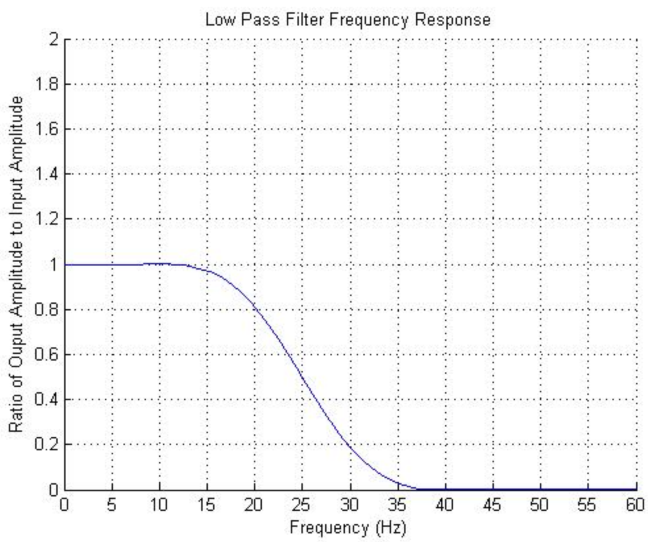


Figure 3

As an example, Figure 3 indicates that if a sinusoidal signal with a frequency of 25 Hz and an amplitude of 1 (so a signal of the form c(t) = 1*sin(2π*25*t)) were to pass through the filter the signal would emerge with a frequency of 25 Hz and an amplitude of 0.50. In the case of our signal x(t) the filter passes the 10 Hz component without affecting its amplitude, and significantly reduces the amplitude of the 50 Hz and 90 Hz components.

It is more convenient to view a frequency response with the y-axis on a log scale as illustrated by figure 3A. Now we see that in the case of our signal x(t) the filter passes the 10 Hz component with a gain $\approx 10^0$ (output amplitude to input amplitude ratio = 1), and reduces the amplitude of the 50 Hz and 90 Hz components by a factor of 10^{-3} (output amplitude to input amplitude ratio = 10^{-3} = 0.001). Note that the low pass filter does not reduce the amplitude of the 50 Hz and 90 Hz components to zero as suggested by Figure 3; the ripples you see in Figure 3A pass a very small fraction of these components to the output (10^{-3} = 0.001 = 0.1% of the amplitude of these components makes it through the filter). We will ignore these ripples in 6.082 and you will study more about them in later courses.

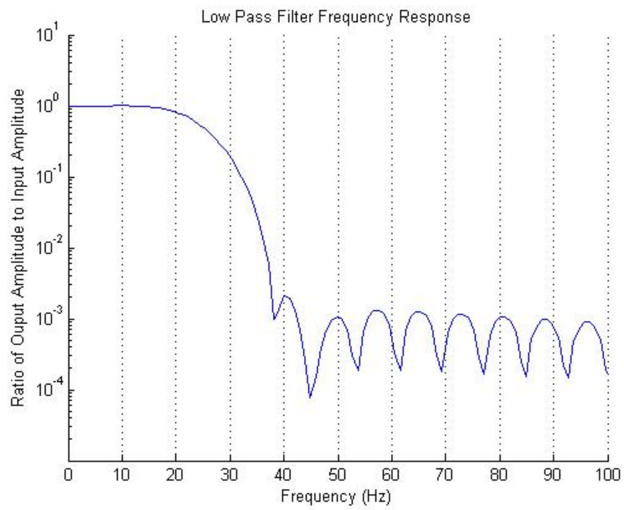


Figure 3A

Figures 4 and 5 illustrate the time and frequency domain representation of the signal x(t) after being filtered by the low pass filter of Figure 3. Note that both in the time and frequency domains there is only evidence of the 10 Hz component, and that this 10 Hz component still has an amplitude of 1. In the time-domain we see 10 oscillations in a period of 1 sec (10 Hz signal), and in the frequency domain we see a single peak at 10 Hz.

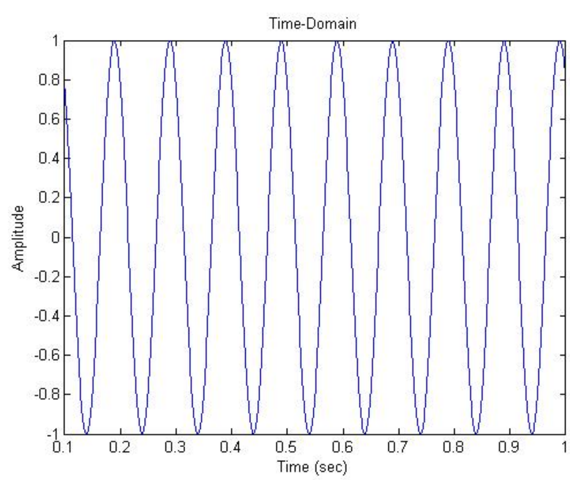


Figure 4

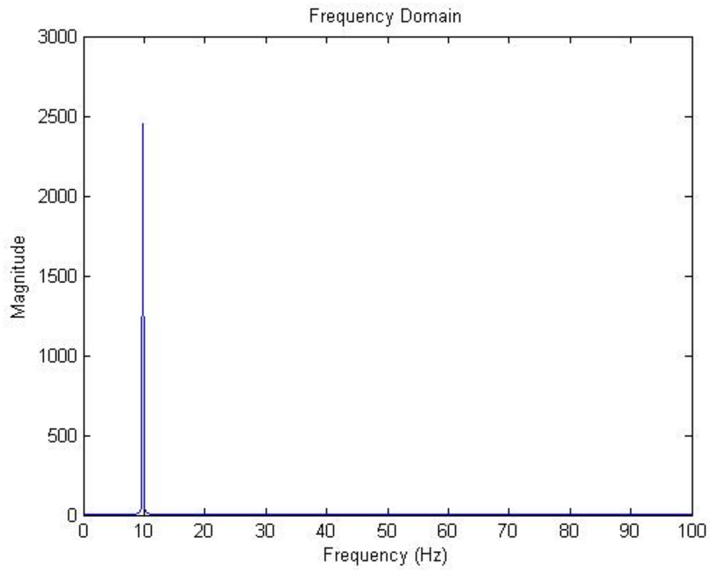


Figure 5

Bandpass Filters

Now consider filtering the input x(t) so that we retain the 50 Hz component and attenuate both the 10 Hz and 90 Hz components. To do this we will use a *bandpass pass filter*. A bandpass filter passes all frequency components falling between the cutoff frequencies $f_{c,low}$ and $f_{c,high}$, and attenuates frequency components lower than $f_{c,low}$ and those greater than $f_{c,high}$. In this example, we will use a bandpass filter with a cutoff frequency $f_{c,low}$ =35 Hz and $f_{c,high}$ =65 Hz in order to pass the 50 Hz component (50 Hz > $f_{c,low}$ and 50 Hz < $f_{c,high}$ so this frequency component passes) and attenuate both the 10 Hz and 90 Hz components (10 Hz < $f_{c,low}$ and 90 Hz > $f_{c,high}$ so these frequency components are attenuated). Note, it is also common to specify a bandpass filter using a *center frequency* and a *bandwidth*; in the previous example the center frequency is 50 Hz and the bandwidth is 30 Hz. Figure 6 and 6A illustrate the frequency response of our bandpass pass filter.

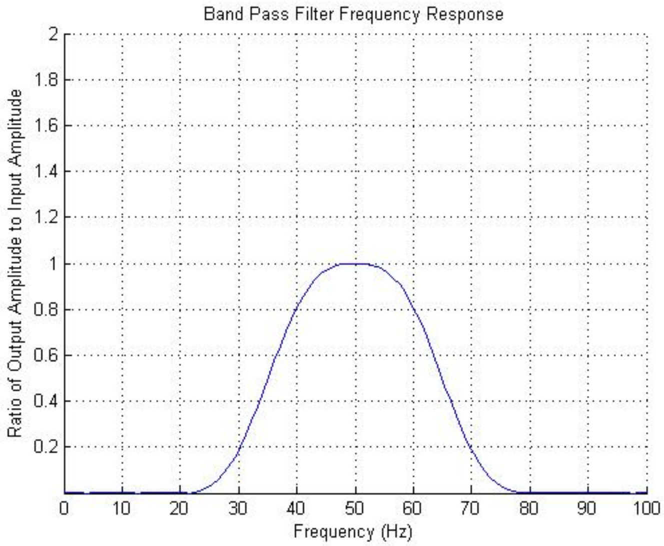


Figure 6

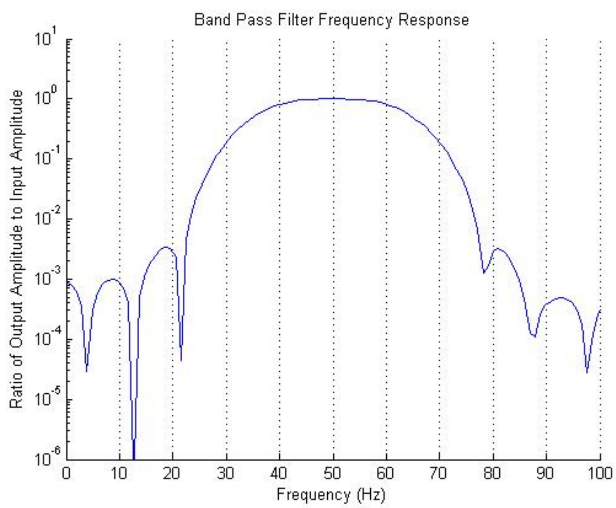


Figure 6A

Figure 6A illustrates that the bandpass filter passes the 50 Hz component with a 10^0 gain (output amplitude to input amplitude ratio = 1), and reduces the amplitude of the 10 Hz and 90 Hz by a factor of 10^{-3} (output amplitude to input amplitude ratio = $10^{-3} = 0.001$).

.

Figures 7 and 8 illustrate the time and frequency domain representation of the signal x(t) after being filtered by the bandpass filter of Figure 6. Note that both in the time and frequency domains there is only evidence of the 50 Hz component, and that this 50 Hz still has an amplitude of 1. In the time-domain we see 10 oscillations in a period of 0.2 sec (50 Hz signal), and in the frequency domain we see a single peak at 50 Hz.

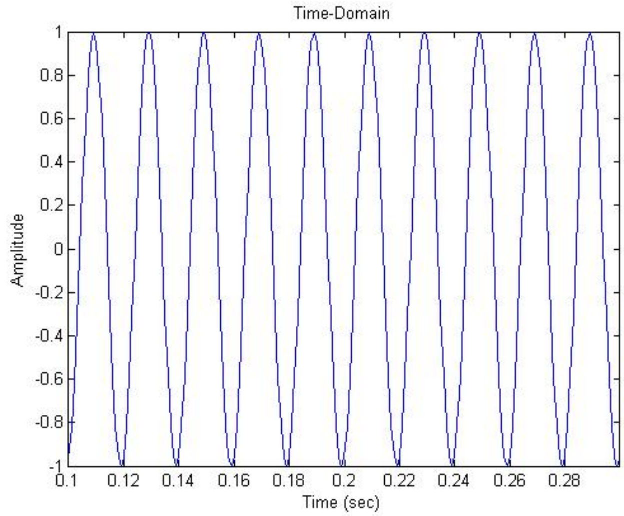


Figure 7

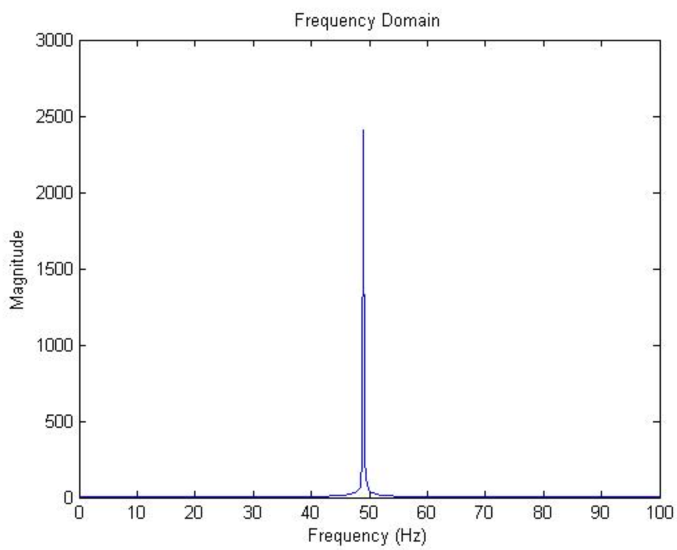


Figure 8

High Pass Filters

Now consider filtering the input x(t) so that we retain the 90 Hz component and attenuate both the 10 Hz and 50 Hz components. To do this we will use a *high pass filter*. A high pass filter passes all frequency components greater than a cutoff frequencies f_c, and attenuates all frequency components lower than f_c. In this example, we will use a high pass filter with a cutoff frequency f_c =70 Hz in order to pass the 90 Hz component (90 Hz > f_c and so this frequency component passes) and attenuate both the 10 Hz and 50 Hz components (10 Hz < 50 Hz < f_c so these frequency components are attenuated). Figure 9 and 9A illustrates the frequency response of our high pass filter.

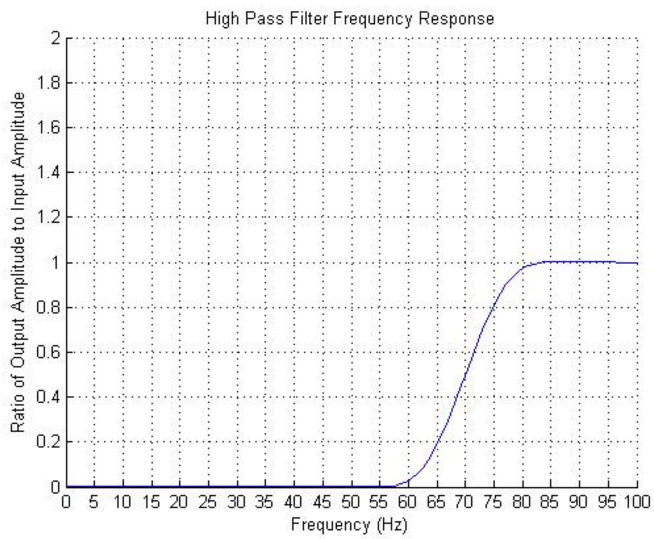


Figure 9

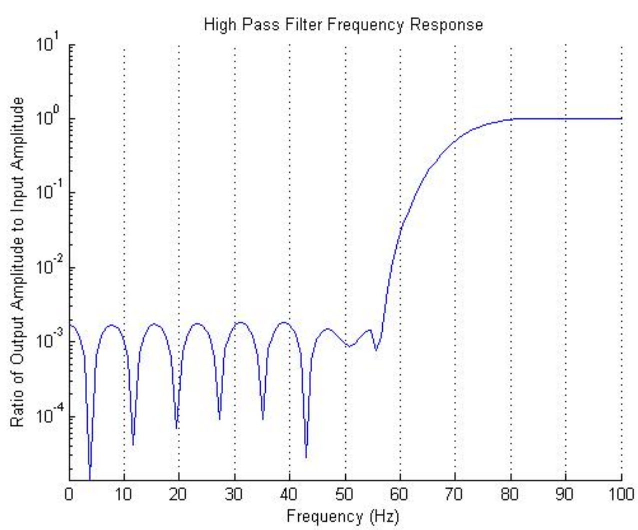


Figure 9A

Figure 9A illustrates that the high pass filter passes the 90 Hz component with a 10^0 gain (output amplitude to input amplitude ratio = 1), and reduces the amplitude of the 10 Hz and 50 Hz components by a factor of 10^{-3} (output amplitude to input amplitude ratio = 10^{-3} = 0.001).

Figures 10 and 11 illustrate the time and frequency domain representation of the signal x(t) after being filtered by the bandpass filter of Figure 9. Note that both in the time and frequency domains there is only evidence of the 90 Hz component, and that this 90 Hz still has an amplitude of 1. In the time-domain we see 9 oscillations in a period of 0.1 sec (90 Hz signal), and in the frequency domain we see a single peak at 90 Hz.

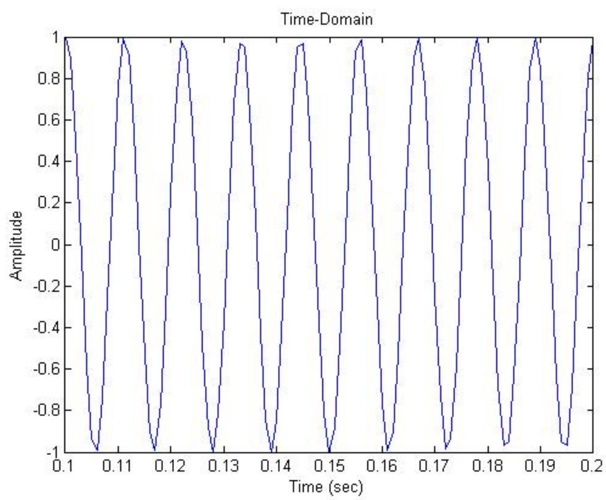


Figure 10

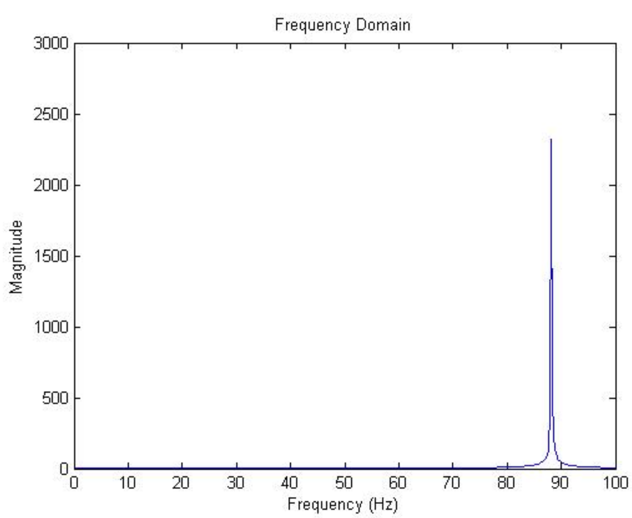


Figure 11

Band Reject Filters

Now consider filtering the input x(t) so that we retain the 10 Hz and 90 Hz components and attenuate the 50 Hz components. To do this we will use a *band reject filter*. A band reject filter behaves in opposite fashion to the bandpass filter; a band reject filter attenuates all frequency components falling between the cutoff frequencies $f_{c,low}$ and $f_{c,high}$, and passes frequency components lower than $f_{c,low}$ and those greater than $f_{c,high}$. In this example, we will use a band reject filter with a cutoff frequency $f_{c,low}$ = 35 Hz and $f_{c,high}$ = 65 Hz in order to pass the 10 Hz and 90 Hz components (10 Hz < $f_{c,low}$ and 90 Hz > $f_{c,high}$ so these frequency components pass) and attenuate the 50 Hz component ($f_{c,low}$ < 50 < $f_{c,high}$ so this frequency component is attenuated). It is again common to specify a band reject filter using a center frequency and a bandwidth; in the previous example the center frequency is 50 Hz and the bandwidth is 30 Hz. Figure 12 illustrates the frequency response of our band reject filter.

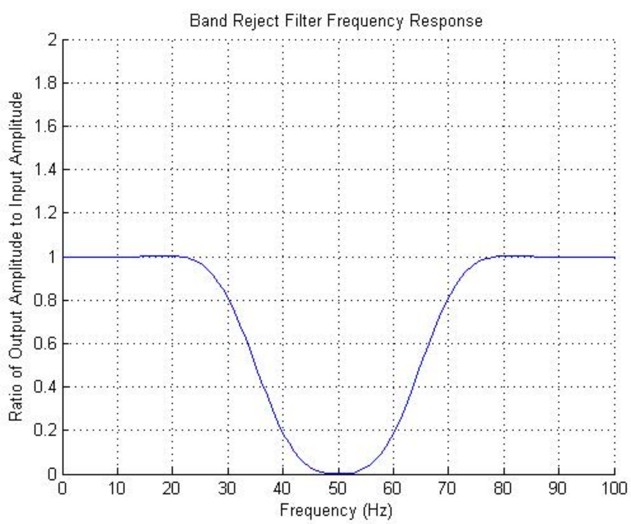


Figure 12

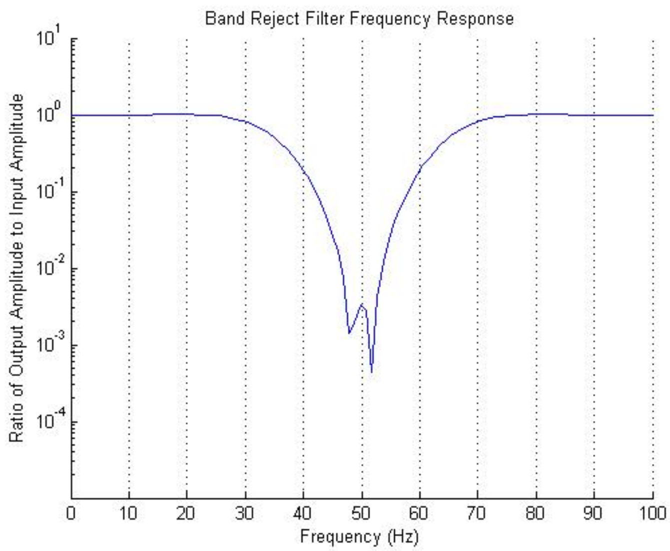


Figure 12A

In the case of our signal x(t) the filter passes the 10 Hz and 90 Hz components without with a 10^0 gain (output amplitude to input amplitude ratio = 1), and reduces the amplitude of the 50 Hz by a factor of 10^{-3} (output amplitude to input amplitude ratio = $10^{-3} = 0.001$). Figures 13 and 14 illustrate the time and frequency domain representation of the signal x(t) after being filtered by the band reject filter of Figure 12. Note that both in the time and frequency domains there is only evidence of the 10 Hz and 90 Hz components.

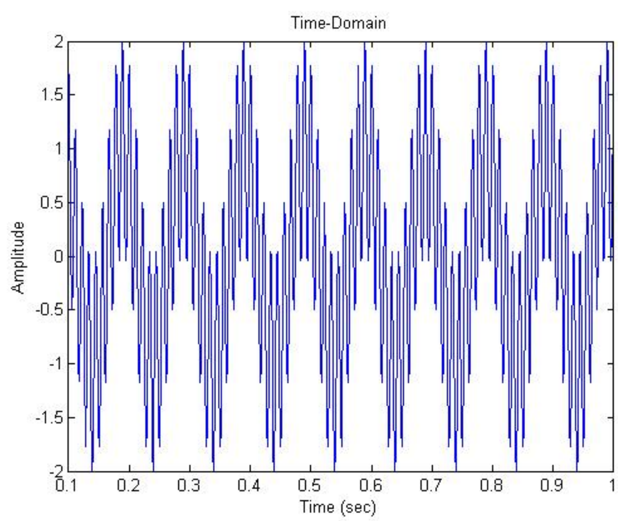


Figure 13

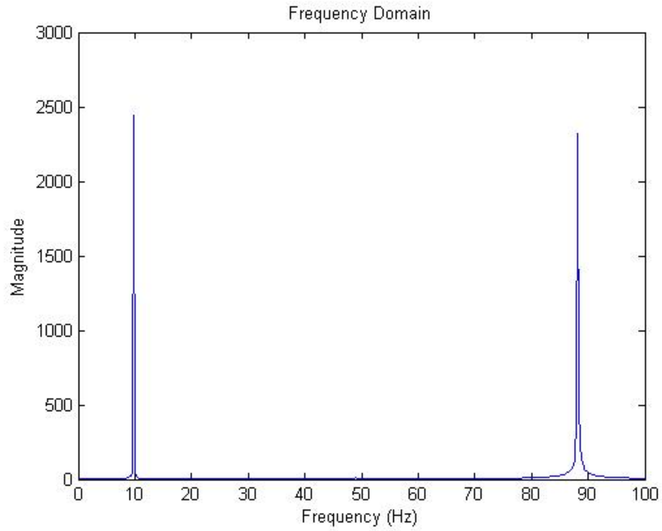


Figure 14

Filtering Signals

At this point you understand how to examine the frequency response of a filter in order to determine how the filter will affect an input signal; now we will explore how to actually filter signals. In the discrete time domain, difference equations are used to filter signals. We rewrite the input x(t) in equation (1) in discrete time by replacing t with n*Ts, where Ts is the sampling time and Fs = 1/Ts.

$$x(n) = \sin(2\pi * 10 * n * Ts) + \sin(2\pi * 50 * n * Ts) + \sin(2\pi * 90 * n * Ts) \qquad (2)$$

In the following sections we illustrate how difference equations behave as filters, and demonstrate how filters such as that in Figure 3 are implemented using difference equations.

Example #1: Difference Equation as Low Pass Filter

Let's begin with an example that demonstrates how a difference equation can behave as a low pass filter. Consider the difference equation below

$$y(n) = \frac{1}{10} * x(n) + \frac{1}{10} * x(n-1) + \frac{1}{10} * x(n-2) + \ldots + \frac{1}{10} x(n-9)$$

$$y(n) = \sum_{i=0}^{9} \frac{1}{10} * x(n-i)$$

Each output sample y(n) is the arithmetic average of the10 input samples x(n), x(n-1), x(n-2), x(n-3), ... , and x(n-9). The averaging operation allows slow changes in the input x(n) (low frequencies) to be reflected in the output signal y(n); in other words, averaging is akin to low pass filtering.

As an example let x(n) to be the input signal defined in equation 1, and let's try to filter x(n) in order to extract the 10 Hz frequency component. Figure 15 and 15A illustrate the frequency response of the filter being implemented by the difference equation.

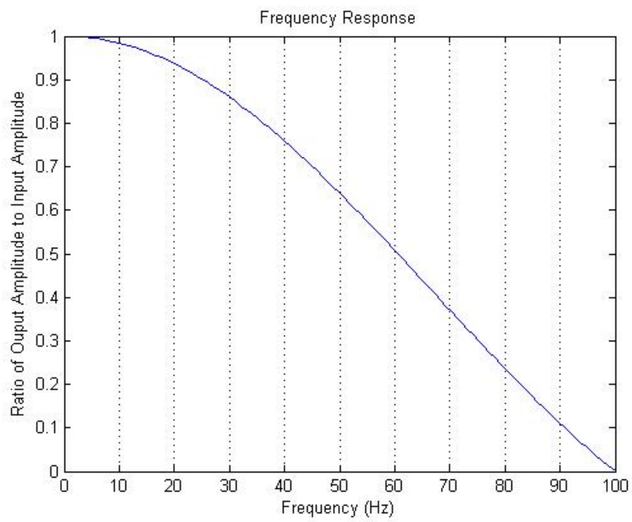


Figure 15

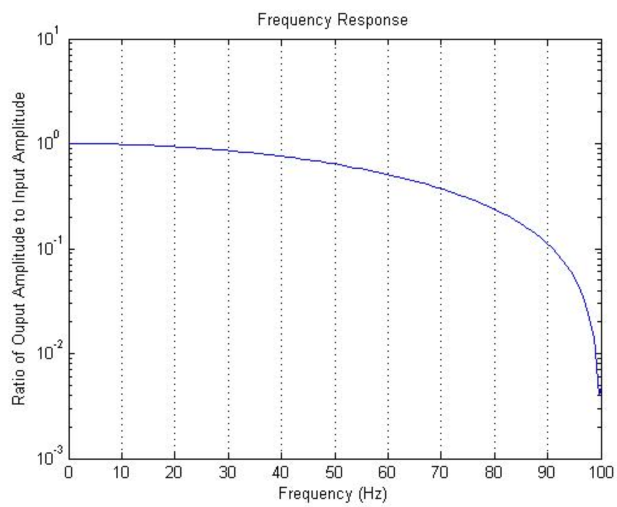


Figure 15A

When x(n) is processed using the difference equation the output y(n) is produced; the frequency spectrum of y(n) is illustrated in Figure 16. Note how the spectral peaks corresponding to the higher frequency components (the 50 Hz and 90 Hz components) have been reduced more than the spectral peak associated with the lower frequency component (the 10 Hz component); the difference equation has attenuated the higher frequency components of x(n) just like a low pass filter.

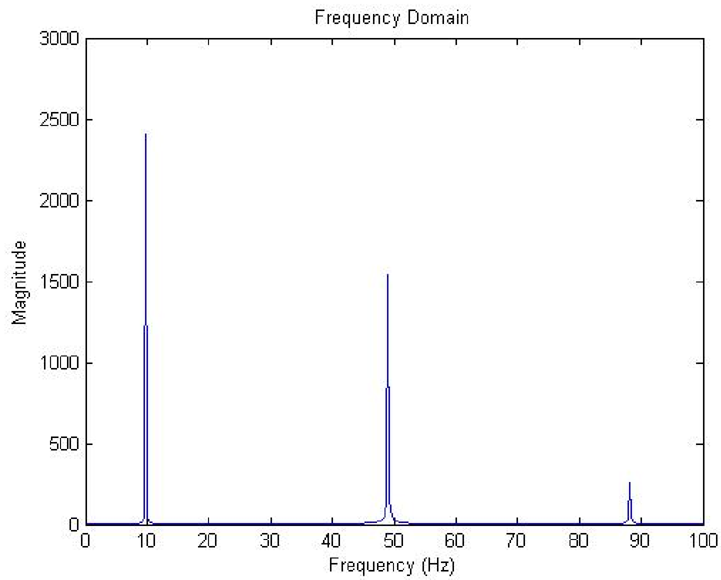


Figure 16

Lets try to increase the number of samples of x(n) that are averaged in the hope of attaining a higher performance low pass filter; in other words, a low pass filter that further attenuates the 50 Hz and 90 Hz. Suppose we now filter x(n) using the difference equation below; the frequency response of the filter implemented by this difference equation is shown in Figure 17 and 17A.

$$y(n) = \frac{1}{35} * x(n) + \frac{1}{35} * x(n-1) + \frac{1}{35} * x(n-2) + \ldots + \frac{1}{35} x(n-34)$$

$$y(n) = \sum_{i=0}^{34} \frac{1}{35} * x(n-i)$$

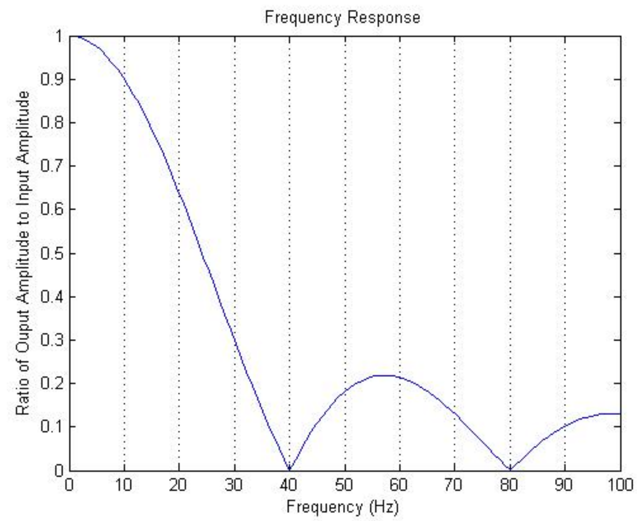


Figure 17

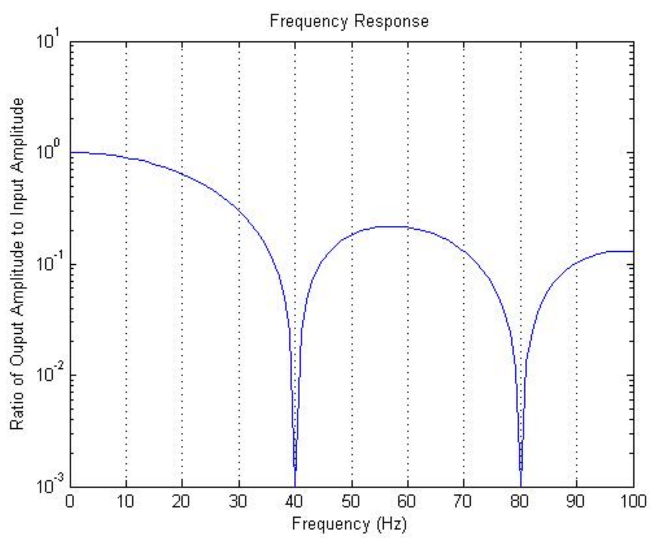


Figure 17A

When x(n) is processed using the difference equation the output y(n) is produced; the frequency spectrum of y(n) is illustrated in Figure 18. Note how we were successful in further reducing the spectral peak associated with the 50 Hz component; however, we also inadvertently reduced the spectral peak of the 10 Hz component. Simply increasing the number of samples of x(n) that are averaged is not a very effective strategy to implementing a high performance low pass filter.

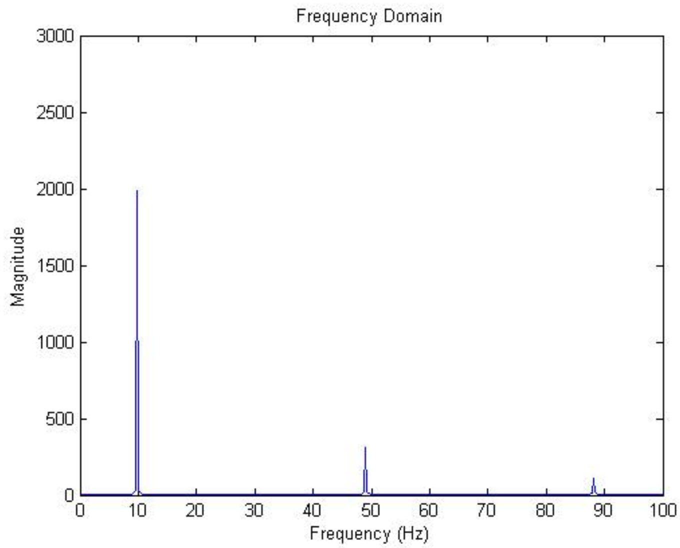


Figure 18

To implement the low pass filter in Figure 3 we averaged 128 of samples of x(n), but not all samples were weighted equally. The weights (or difference equation coefficients) used are plotted in Figure 19. In Figure 19 coefficient number 0 multiplies x(n); coefficient number 1 multiplies x(n-1); coefficient number 2 multiplies x(n-2); and so on. You will learn how to produce these coefficients in Lab #3.

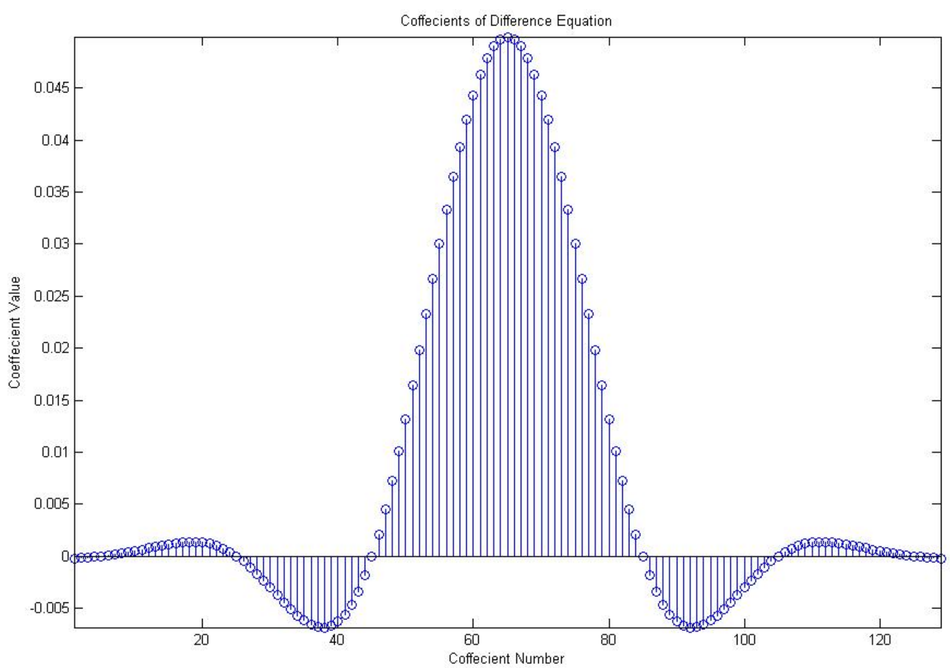


Figure 19

Example #2: Difference Equation as High Pass Filter

Now let's see how a difference equation can behave as a high pass filter. Consider the simple difference equation below

$$y(n) = x(n) - x(n-1)$$

Each output sample y(n) is the difference of the two input samples x(n), and x(n-1). The difference operation prevents slow changes or constants in the input x(n) (low frequencies) from being reflected in the output signal y(n); in other words, the difference operation is akin to high pass filtering.

As an example, let x(n) be the input signal defined in equation 1. Compare the spectrum of x(n) (shown in Figure 2) with the spectrum of y(n) (shown in Figure 20). Note how the spectral peaks corresponding to the 10 Hz and 50 Hz components have been reduced more than the peak corresponding to 90 Hz component; the difference equation has attenuated the lower frequency components of x(n) just like a high pass filter.

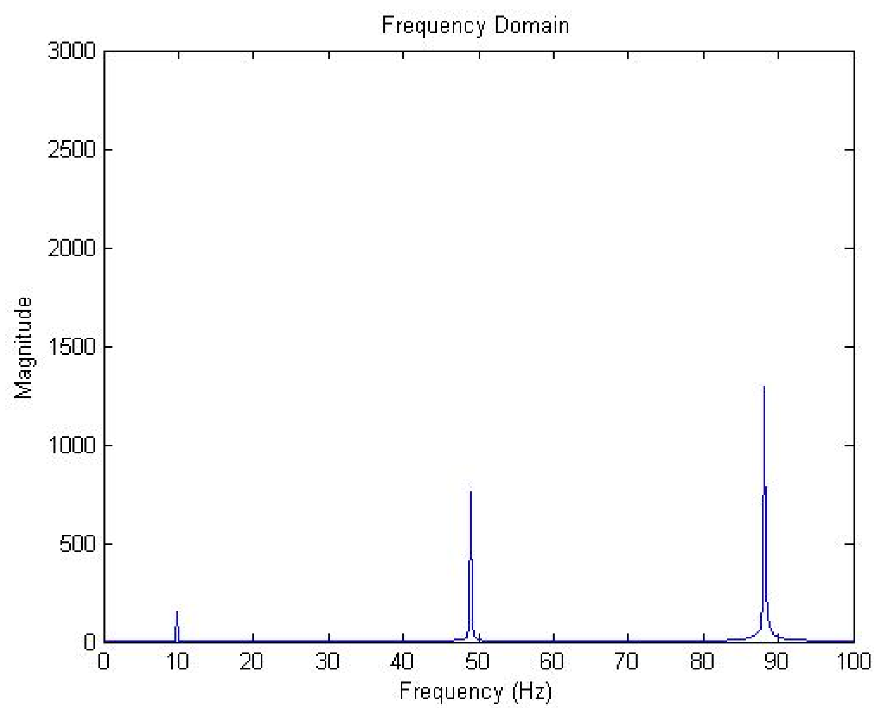


Figure 20

Custom Design Filters Using Difference Equation

If you examine carefully the difference equations in Examples 1 and 3 you will see they fit the general pattern expressed below

$$y(n) = b_0 x(n) + b_1 x(n-1) + b_2 x(n-2) + \ldots + b_N x(n-N)$$
$$y(n) = \sum_{i=0}^{N} b_i x(n-i)$$

In the case of the low pass filter, we used ten b_i coffecients and they were all equal to 1/10. In the case of the high pass filter we had two b_i coefficients equal to 1 and -1 respectively. *The value of the b_i coefficients and the number of these coefficients determines the type of filter the difference equation implements.* The number of coffecients is also known as the *filter order.*

The mathematics involved in picking the b_i coefficients to implement a particular filter will be discussed in more advanced courses, but you will learn in Lab #3 how to use Matlab to produce these coefficients. You are ready to appreciate how the order of the difference equation changes the filter properties.

Figure 21 and 21A shows the frequency response of bandpass filters with cutoff frequencies $f_{c,low}$ =25 Hz and $f_{c,high}$ =75 Hz implemented using difference equations with orders of N=64, 128, and 4096. With increasing order the filter improves in its ability to pass frequencies between $f_{c,low}$ and $f_{c,high}$, and attenuates those frequencies outside this range.

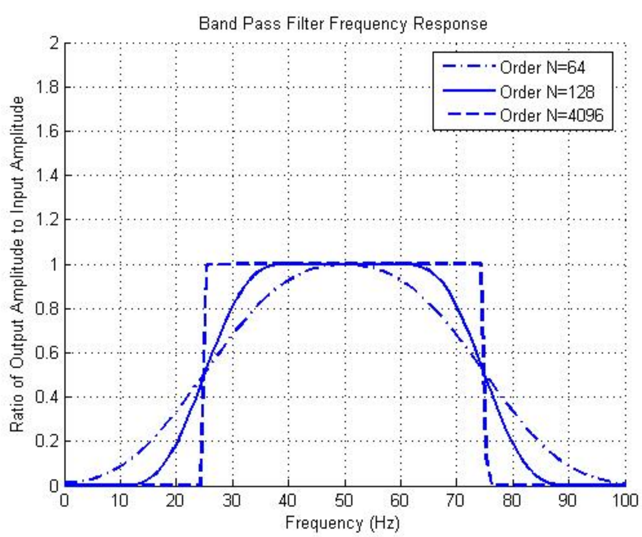


Figure 21

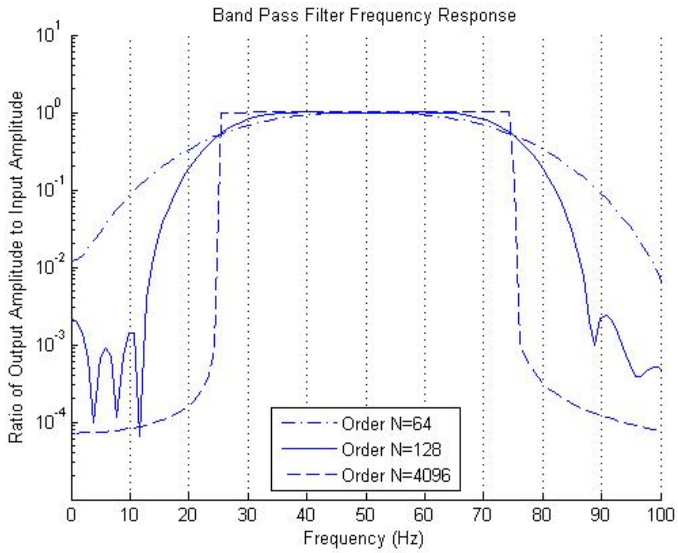


Figure 21A

As an example consider how each filter passes a signal component with a frequency of f=70 Hz. The lower order filters (N=64,128) attenuate this frequency component even though $f_{c,low} < f < f_{c,high}$, but the higher order filter (N=4096) passes this component with negligible attenuation. Now consider how each filter attenuates a signal component with a frequency of f=80 Hz. The lower order filters (N=64,128) pass this frequency even though $f > f_{c,high}$, but the higher order filter (N=4096) severely attenuates this component.

There is a **cost** associated with using high-performance filters with large orders. The number of computations involved to produce a single output sample y(n) using the difference equation is 2N-1 (N multiplications and N-1 summations). The greater the number of computations the longer the *processing time* and *processing power* required by the electronic device using the filter. This is of great concern for designers of low-power, mobile electronic devices such as cellular phones.

General Form of Difference Equation

The output of a difference equation need not only depend on present and past values of the input x(n), it can also depend on past values of the output y(n). Generally, a difference equation takes the following form

$$y(n) = a_1 y(n-1) + a_2 y(n-2) + \cdots a_M y(n-M) + b_o x(n) + b_1 x(n-1) + b_2 x(n-2) + \ldots + b_N x(n-N)$$

$$y(n) = \sum_{j=1}^{M} a_i y(n-j) + \sum_{i=0}^{N} b_i x(n-i)$$

.

As was the case in the previous section, the values of the a_j and b_i coefficients as well as their numbers M and N respectively (M need not equal N) determines the type of filter implemented by the difference equation. Consider the following difference equation as an example

$$y(n) = 0.85 * y(n-1) + 0.073 * x(n) + 0.073 * x(n-1)$$

Each output sample y(n) is a linear combination of the input samples x(n) and x(n-1), and the output sample y(n-1). Let x(n) be the input signal defined in equation 1. Compare the spectrum of x(n) (shown in Figure 2) with the spectrum of y(n) (shown in Figure 22). Note how the spectral peaks corresponding to the higher frequency components (the 50 Hz and 90 Hz components) have been reduced; this difference equation has attenuated the higher frequency components of x(n) just like a low pass filter. Figures 23 and 23A show the frequency response of the low pass filter implemented using this difference equation.

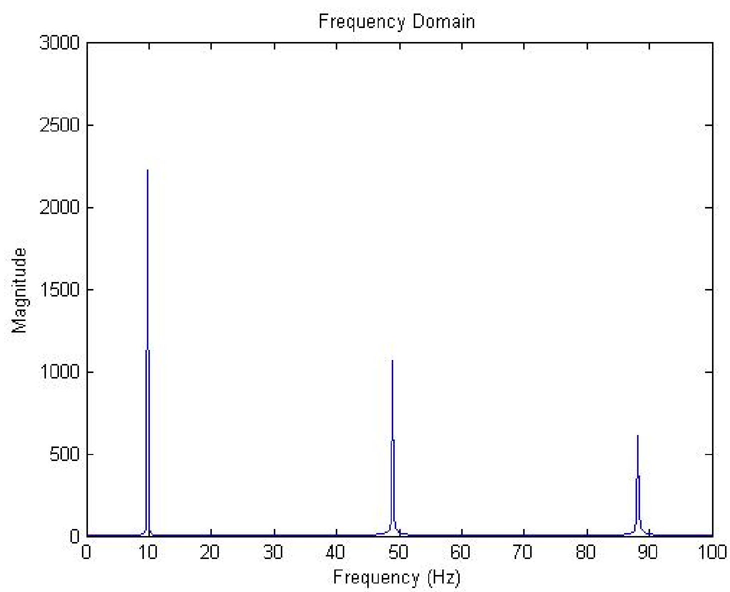


Figure 22

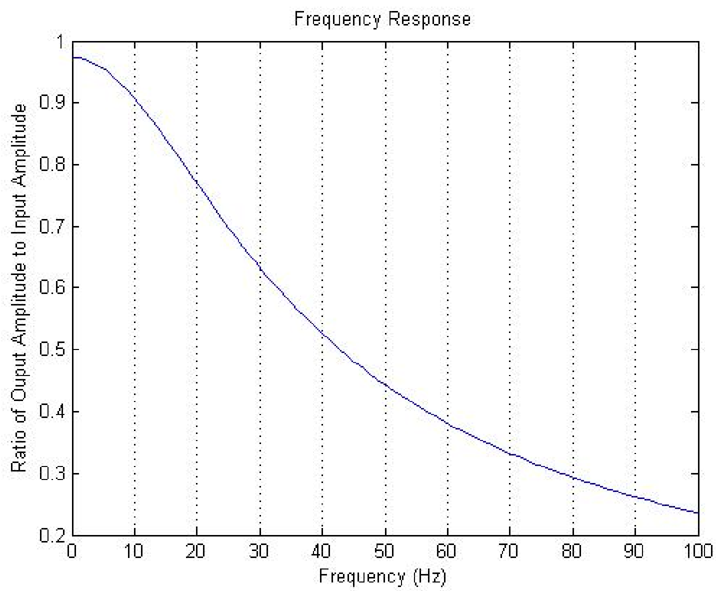


Figure 23

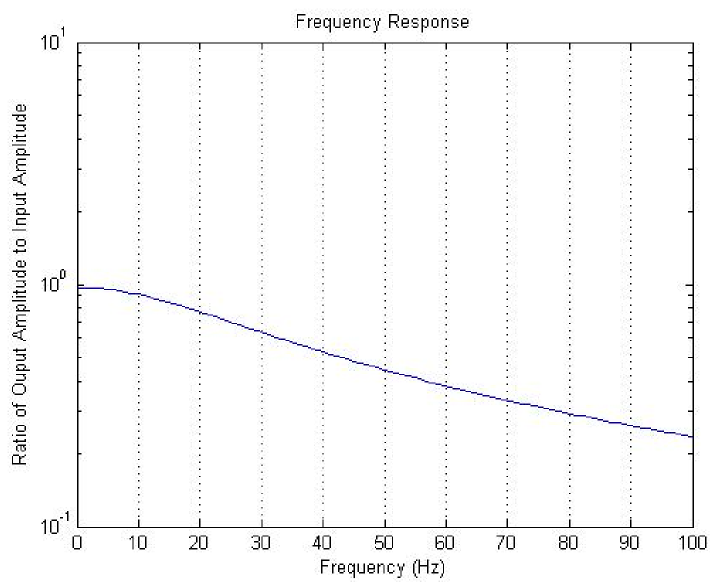


Figure 23A