EXPERIMENT 9

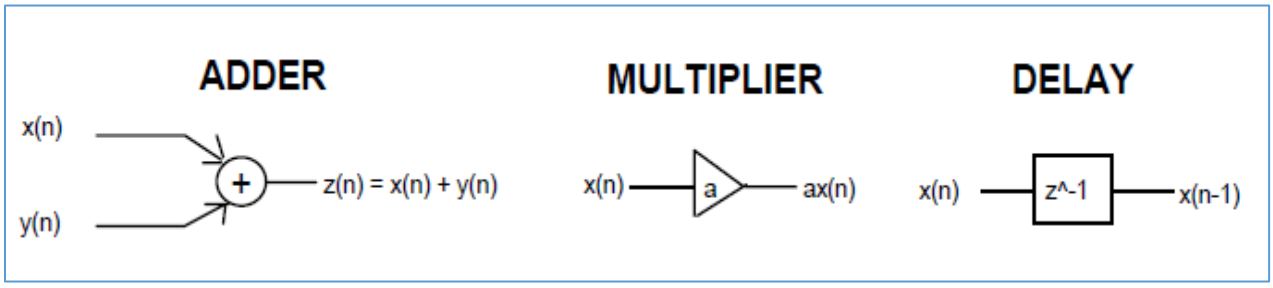
Digital Filters

Objective

To study the method of giving delay in a signal and then adding or multiplying it with another signal such that data of both signals remain intact. Moreover, to understand chebyshev, butterworth and elliptic techniques for obtaining higher, lower or a band of frequencies of a signal.

Introduction

A fundamental aspect of signal processing is filtering. Filtering involves the manipulation of the spectrum of a signal by passing or blocking certain portions of the spectrum, depending on the frequency of those portions. Filters are designed according to what kind of manipulation of the signal is required for a particular application. Digital filters are implemented using three fundamental building blocks: an adder, a multiplier, and a delay element.



With the basic building blocks at hand, the two different filter structures can easily be implemented. These two structures are Infinite Impulse Response (IIR) and Finite Impulse Response (FIR), depending on the form of the system's response to a unit pulse input. IIR filters are commonly implemented using a feedback (recursive) structure, while FIR filters usually require no feedback (non-recursive).

Using MATLAB, a low pass digital filter is designed using various analog prototypes: Chebyshev, Butterworth, and Elliptic. The optimum filter type is chosen on the basis of implementation complexity, magnitude response, and phase response.

Chebyshev Filter

[b,a] = cheby1(n,R,Wp,*'ftype'*) designs a highpass, lowpass, or bandstop filter, where the string *'ftype'* is 'high', 'low', or 'stop'.

n is the order of filter. W_PIs normalized pass-band edge frequency. $R$is the p-p ripple, usually 0.5dB.

MATLAB Code Chebyshev:

```
wp = 0.125*2*pi; % digital passband frequency in Hz (normalized)
ws = 0.1375*2*pi; % digital stopband frequency in Hz (normalized)
Rp = 0.5; % passband ripple in dB
As = 20; % stopband attenuation in dB

[b, a] = cheby1(8, Rp,wp, 'low');
[db,w] = freqz(b,a);
plot(w/pi,abs(db));
xlabel('frequency (Hz)'); ylabel('decibels'); title('Magnitude in dB');
```

Exercise 1:

Run the code given above for low and high pass filter prototype and show the absolute and phase plots on MATLAB. Attach the screen shot of the plots in the space provided below:

Butterworth Filter

[b,a] = butter(n,Wn,*'ftype'*) designs a highpass, lowpass or bandstop filter, where the string *'ftype'* is 'high', 'low', or 'stop', as described below.

n is the order of filter and W_n is normalized cutoff frequency.

MATLAB Code Butterworth:

```
wp = 0.125*2*pi; % digital passband frequency in Hz (normalized)

ws = 0.1375*2*pi; % digital stopband frequency in Hz (normalized)

Rp = 0.5; % passband ripple in dB

As = 20; % stopband attenuation in dB


[b, a] = butter(8,wp, 'low');

[db,w] = freqz(b,a);

plot(w/pi,abs(db));

xlabel('frequency (Hz)'); ylabel('decibels'); title('Magnitude in dB');
```

Exercise 2:

Run the code given above for low and high pass filter prototype and show the absolute and phase plots on MATLAB. Attach the screen shot of the plots in the space provided below:

Elliptic Filter

[b,a] = ellip(n,Rp,Rs,Wp,*'ftype'*) designs a highpass, lowpass, or bandstop filter, where the string *'ftype'* is 'high', 'low', or 'stop'.

Here, $n$is the order of filter. $W_P$is normalized pass-band edge frequency. R_P is the p-p ripple, usually 0.5dB.
NB: stop-band ripple must be greater than pass-band ripple.

MATLAB Code Elliptical

```
wp = 0.125*2*pi; % digital passband frequency in Hz (normalized)

ws = 0.1375*2*pi; % digital stopband frequency in Hz (normalized)

Rp = 0.5; % passband ripple in dB

As = 20; % stopband attenuation in dB


[b, a] = ellip(5,Rp,1.0,wp, 'low');

[db,w] = freqz(b,a);

plot(w/pi,abs(db));

xlabel('frequency (Hz)'); ylabel('decibels'); title('Magnitude in dB');
```

Exercise 3:

Run the code given above for low and high pass filter prototype and show the absolute and phase plots on MATLAB.

Distinguish between the responses of the three filters.

Post Lab:

Design a Band-pass Butterworth filter of order 32. Choose the band frequencies yourself. Show the absolute and phase plots and your code for this Band-pass filter in MATLAB .Attach the screen shot of the code.