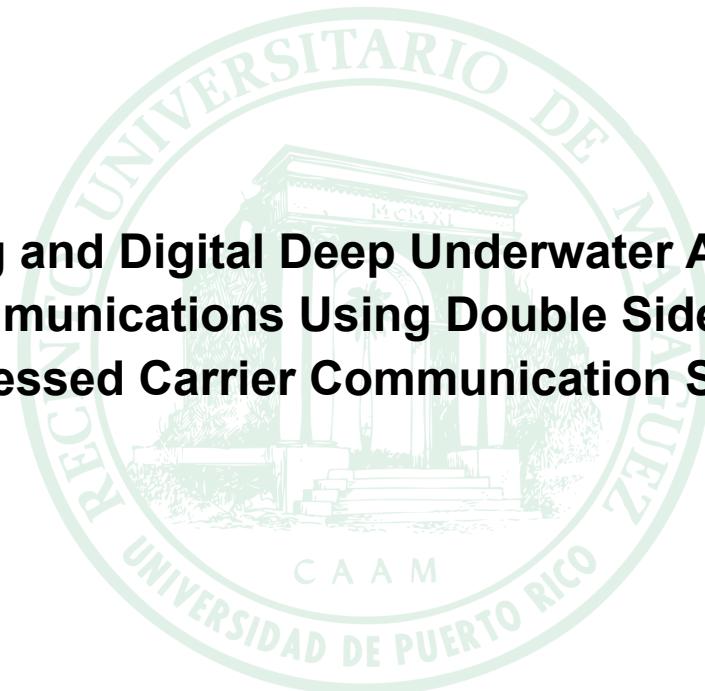


University of Puerto Rico at Mayagüez  
Department of Computer and Electrical Engineering

# **Analog and Digital Deep Underwater Acoustic Communications Using Double Sideband Suppressed Carrier Communication Systems**



**INEL 4301 Sec. 096**  
**Theory of Communication**  
**Grupo #1 compuesto por:**  
Roberto Sanchez  
Carlos Del Valle  
Javier Pagan  
Genesis Borges  
Osvaldo Aquino  
Jeziel Torres  
**Profesor:** Dr. Domingo Rodriguez  
**December 4, 2021**

## A. Project Description

In this project we designed a communication system that uses a "double sideband suppressed carrier" (DSB-SC). The idea is to use this system to carry out digital and analog communication through an underwater medium. The tasks of this project involve the implementation and modeling of the different components that surround digital and analog communications in an DSB-SC system, from the input of wanted and unwanted signals, to the processing and modifications of these signals to get an output that is approximately equal to the input. This project takes as input a "Dual Tone Multi Frequency" (DTMF) signal, transmits it through an underwater medium, and finally receives it.

### Part 1

1. Design an ideal, linear phase, band-pass filter to eliminate the input interference signal  $g(t)$ . Important figures of merit to rate this task will be center frequency, filter's order, and filter's bandwidth.
2. Design an ideal, linear phase, low-pass filter to model the channel filter  $TC$ . Important figures of merit to rate this task will be filter's bandwidth and filter's order.
3. Modify the power of the noise signal  $n(t)$  and describe the computed channel's output signal/noise ratio (SNR) characteristics. For this task, assume the SNR at the channel's input is equal to 80 dB. Important figures of merit to rate this task will be the lowest SNR obtained at the channel's output, allowing intelligibility of the wanted output signal, as well as the average and variance values of the AWGN.
4. Design an ideal, linear phase, band-pass filter  $TR$  to reduce the receiver's noise and improve the signal/noise ratio at the input of the demodulator's subsystem when compared to the signal/noise ratio at the output of the channel; that is, by comparing the spectral densities between the channel

output signal  $y_{co}(t)$  and the demodulator input signal  $y_{di}(t)$ . Important figures of merit to rate this task will be center frequency, filter's order, filter's bandwidth, as well as the highest SRN obtained at the filter's output.

5. Design a linear phase, low-pass filter, with cut-off frequency  $FV = 2000\text{Hz}$  for the receiver's filter  $TL$ . The most important figure of merit for this task will

be the lowest time-delay,  $D_r = (\frac{M_L - 1}{2})T_s$ , obtained for this filter, where  $M_L$  is the order of the filter and  $T_s = \frac{1}{F_s}$  is the overall sampling time of the system.

6. Design an ideal, linear phase, band-pass filter (upper sideband) to eliminate the input interference signal  $g(t)$ . Proceed to leave all the designed subsystems without modifications, except for the band pass filter at the transmitter which will be modified and substituted by this new upper sideband band pass filter. Important figures of merit to rate this task will be the filter's order.

## Part 2

1. Proceed to quantize the modulating signal  $x_m(t)$  used in Part 1 of the project. Record the output signal  $x_{mq}(t)$  of the quantizer. A figure of merit for this task will be the lowest number of quantization levels used. Record intelligible  $x_r(t)$ .

2. Proceed to quantize the desired or wanted signal  $s(t)$  and the interference or unwanted signal  $g(t)$ , separately. Use this added quantized signal  $x_{mq}(t)$  as a new modulating signal for the system, repeating Task 2.1, above.

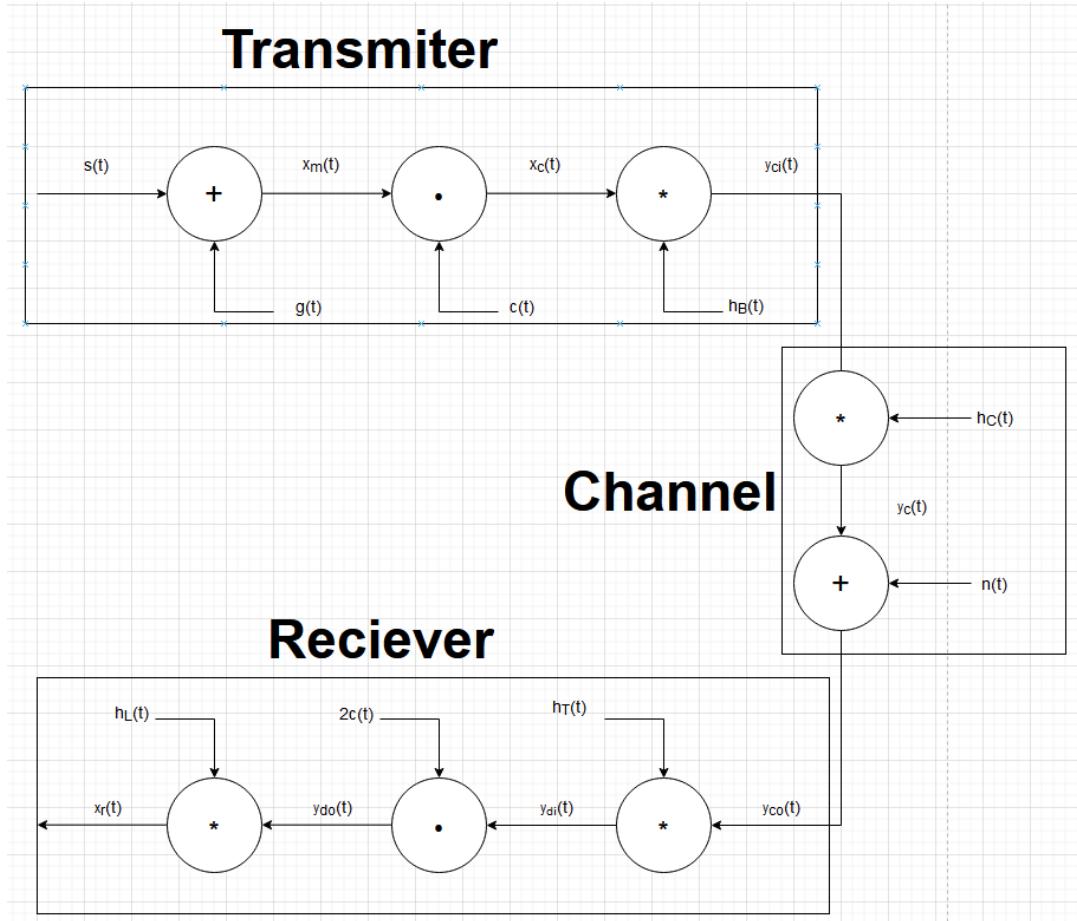
## B. Theoretical Formulation

To advance this communication system we take as a basis some theoretical issues. In the first part, we have to transmit the DTMF signal using analog communication. It is assumed that the maximum frequency content of the DTMF is about 2000Hz, although the highest signal in a DTMF is 1633Hz. The channel is composed of a basic ideal, linear phase, low-pass filter with a one sided bandwidth of 12000Hz, along with a summing system which is added to the output of the aforementioned filter.

For the design of the system, it was necessary to assign values to various variables. The Nyquist-Shannon theorem was key to the calculation of the sampling frequency, or Sampling ( $F_0$ ). For this calculation, it was taken into account that the carrier frequency ( $f_c$ ) should belong to a predefined range, between 6000Hz and 10000Hz. The carrier frequency value was chosen to be 8000Hz, since it is the midpoint between these two numbers. Finally, using this value and the value for the maximum frequency content of the DTMF signal, it was calculated using the Nyquist-Shannon theorem  $F_s > 2(2f_c + F_v)$  that the sampling frequency must be above 36000, thus a value of 40000Hz was chosen.

The input signal to the modulator was the desired signal  $x_m(t)$ , which is composed of  $s(t)$ , the wanted DTMF signal, and of an unavoidable and unwanted signal (interference) called  $g(t)$ . This signal is then passed through bandpass filter which tries to filter out all frequencies below  $f_c - F_v$  and all signals above  $f_c + F_v$ , thus removing  $g(t)$ . This signal goes through the channel, which is composed of a low-pass filter and additive white noise. Since the one sided channel bandwidth is 12000Hz, and since our signal is between 6000Hz and 10000Hz, the information would ideally not get affected. Afterwards, an additive white noise is added, increasing the noise content of

our desired signal. This signal is once again passed through a bandpass filter, removing all noise aside from that which was in the same frequency as our desired signal. Finally, the signal passes through the demodulator, and then passes through a low-pass filter (TL) with a defined cutoff frequency to recover the desired signal. This process is illustrated in Figure 1.



**Figure 1:** DSB-SC communication system's basic architecture

The signal to noise ratio (SNR) is a value that compares the power of the desired signal and of the noise. In decibels, a higher value means that the desired signal is more present than the noise. In part 1, task 3, we have to describe the computed channel SNR while varying the power of  $n(t)$ , the noise inside of the channel. Assuming that the channel's input's SNR is 80dB, then the SNR at the output would be

$$SNR_{output} = SNR_{input} - SNR_{n(t)} = 80dB - SNR_{n(t)}. \text{ Since the variance}$$

dictates the value of the SNR of the noise, this equation would be

$$SNR_{output} = 80dB - 20\log\left(\sqrt{\sigma^2}\right).$$

Quantization is the process of turning a potentially infinite set of values into a discrete set of values. For example, a function that turns all positive numbers to 1 and all negative numbers to 0 is a quantizer, and since it only has 2 possible values it would be a 1 bit quantizer. Finding the amount of bits needed to quantize a signal without losing a significant portion of the signal is an important step in digitizing a signal.

### C. Result Discussion

Before performing the tasks, the input signals to be used in the system were created. The  $n(t)$ ,  $g(t)$  and DTMF or  $s(t)$  signals were created and exported to audio files (.wav) using Matlab. The signal  $x_m(t)$  that represents the sum of the desired and unwanted signals ( $s(t) + g(t)$ ) was created and exported in a similar way.

Starting with Part I, in the first task, a band-pass filter was designed to keep  $s(t)$  (Figure 2) and to eliminate the interference signal  $g(t)$  (Figure 3) at the input. For this, the maximum frequency content and the carrier frequency determined the band limits. The filtered signal becomes the input signal to channel  $Y_{ci}(t)$ . Since the carrier frequency was set to be 8000Hz (Figure 4), and since the interference was chosen to have frequencies of 4800Hz, 5200Hz, and 5800Hz, when the modulating signal got modulated, it caused no problems since the lowest frequency did not pass 0Hz. Additionally, because the maximum frequency content of the DTMF signal was approximately 2000Hz, the values for bandpass filter's bandwidth could readily be chosen to be the carrier frequency plus and minus 2000Hz. The order of the filter dictates how close to ideal the filter would act, the higher the order, the faster the impulse went from 0 to 1 and fell from 1 to 0. It was chosen to have a filter order of 100 since above that there were no drastic changes in efficiency.

In the second task, a low-pass filter was designed to model the channel filter (TC) which is the filter which produces the signal that passes through the channel, called  $Y_c(t)$ . As mentioned above, the order of the filter is 100. Its bandwidth was predetermined by the project description at 12000 Hz. Since all current wanted information would be below 10000Hz, the bandwidth would not distort the desired signal.

In the third task, the channel noise signal, named  $n(t)$ , was created. It represents the noise from the channel, the underwater medium, and adds to the signal that passes through the channel. This is built using the Matlab randn(n) function. This function chooses a pseudorandom with a mean of 0 and a standard deviation of 1. To change its variance manually, it is multiplied by the square root of the variance. It was observed that as the variance lowers, the SNR at the output increases. This made sense, since if on average the value was 0, and if the value does not vary by much, then the noise power would be much lower than that of the desired signal. According to research on signal intelligibility, intelligibility is achieved by obtaining an SNR of 12 dB or more. This means that if the input signal was 80 dB, as long as the total noise power does not exceed 68dB, the wanted signal should be intelligible. It was calculated that the SNR at the output was around 29dB, and since the system had a theoretical maximum of 34.7dB (since the signal  $s(t)$  was approximately 34.7dB) the signal would mostly be formed by the wanted signal and would be intelligible. Increasing the variance vastly lowered the SNR, for example a variance of 2 would lower the SNR to -5.7dB, and would make the output an unintelligible noise.

Once the signal entered the receiver, the signal would be passed through a bandpass filter. This bandpass filter was there to help remove some of the noise from the channel. This filter acted almost exactly like the bandpass filter from the transmitter. The highest SNR obtained was with an order of 900, and it was approximately 30.0164, although this varied greatly depending on the noise from the channel.

This signal would then get demodulated, and would finally enter a low pass filter. This filter's function is to remove the images of the wanted signal's frequency which would be present at the carrier frequency. Since the maximum frequency content of the DTMF signal was 2000Hz, the low pass filter could be given a bandwidth of 2000Hz. The important value though is the time delay caused by the filter, which could get calculated as

$$D_r = \left(\frac{M_L - 1}{2}\right)T_s.$$

Since the sampling time is constant, the time delay would be directly proportional to  $M_L$ . Thus, decreasing the order of the filter would make the process faster, but it would also make the filter less ideal. Since time is not an issue in this project, it is not necessary to have a low order to have a small time delay, although other projects could opt to have lower orders if low delays are important.

Finally, the sixth task changed the band-pass filter on the transmitter to an upper sideband band-pass filter. This means that the negative portion of the DTMF's frequency spectrum would get eliminated. This did not cause any significant changes to the SNR or to any other parts of the system. This means that if it were easier to design a band-pass filter with a smaller bandwidth, it could get done by only filtering the upper or lower sideband of the frequency spectrum.

In part 2 the same DTMF signals were transmitted through a DSB-SC, but in a digital format. This is done by quantizing the modulating signal, which means turning the infinite set of values which the modulating signal could take and representing them with a finite number of values. It was necessary to quantize the modulating signal to at least 3 discrete levels to conserve intelligibility. If 2 levels of discretizations were used, the received signal would sound like quiet white noise.

Finally, instead of quantizing the modulating signal, the last part required quantizing the wanted signal and the interference separately, and keeping the rest of the system the same. The system also required at least 3 levels of discretization, although at that level it was also more intelligible than that of the previous section.

**D. Task Assignment Table**

	Task. 1 Value	Task. 2 Value	Task. 3 Value	Task. 4 Value	Task. 5 Value	Task. 6 Value	Task. 7 Value	Task. 8 Value	
	10	10	10	10	15	15	15	15	
Del Valle, Carlos	100, 10	XXX, YY	100, 10	XXX, YY	XXX, YY	XXX, YY	XXX, YY	XXX, YY	20
Torres, Jeziel	XXX, YY	100, 10	XXX, YY	100, 10	XXX, YY	XXX, YY	XXX, YY	XXX, YY	20
Borges, Génesis	XXX, YY	XXX, YY	XXX, YY	XXX, YY	XXX, YY	100, 15	XXX, YY	XXX, YY	15
Sánchez, Roberto	XXX, YY	XXX, YY	XXX, YY	XXX, YY	XXX, YY	XXX, YY	100, 15	XXX, YY	15
Pagán, Javier	XXX, YY	XXX, YY	XXX, YY	XXX, YY	100, 15	XXX, YY	XXX, YY	XXX, YY	15
Aquino, Osvaldo	XXX, YY	100, 15	15						

100	100	100	100	100	100	100	100
-----	-----	-----	-----	-----	-----	-----	-----

## E. References

- [1] "Single-sideband modulation," *Wikipedia*, 10-Jul-2021. [Online]. Available: [https://en.wikipedia.org/wiki/Single-sideband\\_modulation](https://en.wikipedia.org/wiki/Single-sideband_modulation). [Accessed: 03-Dec-2021].
- [2] "Signal-to-noise ratio - Wikipedia", En.wikipedia.org, 2021. [Online]. Available: [https://en.wikipedia.org/wiki/Signal-to-noise\\_ratio](https://en.wikipedia.org/wiki/Signal-to-noise_ratio). [Accessed: 03- Dec- 2021].
- [3] "Write audio file - MATLAB audiowrite- MathWorks América Latina", La.mathworks.com, 2021. [Online]. Available: <https://la.mathworks.com/help/matlab/ref/audiowrite.html>. [Accessed: 04- Dec- 2021].
- [4]"Estimación de densidad espectral de potencia de Periodogram - MATLAB periodogramMathWorks América Latina", La.mathworks.com, 2021. [Online]. Available: <https://la.mathworks.com/help/signal/ref/periodogram.html>. [Accessed: 04- Dec- 2021].
- [5]"Find the mean value of a signal - MATLAB Answers - MATLAB Central", La.mathworks.com, 2021. [Online]. Available: <https://la.mathworks.com/matlabcentral/answers/673833-find-the-mean-value-of-a-signal>. [Accessed: 03- Dec- 2021].
- [6]"Quantization (signal processing) - Wikipedia", En.wikipedia.org, 2021. [Online]. Available: [https://en.wikipedia.org/wiki/Quantization\\_\(signal\\_processing\)#Analog-todigital\\_converter](https://en.wikipedia.org/wiki/Quantization_(signal_processing)#Analog-todigital_converter). [Accessed: 03- Dec- 2021].
- [7]"Variance - MATLAB var- MathWorks América Latina", La.mathworks.com, 2021. [Online]. Available: [https://la.mathworks.com/help/matlab/ref/var.html#:~:text=V%20%3D%20var\(%20A%20%2C%20w%20%2C%20vecdim%20\)%20computes%20the,by%20dimensions%201%20and%202](https://la.mathworks.com/help/matlab/ref/var.html#:~:text=V%20%3D%20var(%20A%20%2C%20w%20%2C%20vecdim%20)%20computes%20the,by%20dimensions%201%20and%202). [Accessed: 04- Dec- 2021].
- [8]"Intelligibility (communication) - Wikipedia", En.wikipedia.org, 2021. [Online]. Available: [https://en.wikipedia.org/wiki/Intelligibility\\_\(communication\)#cite\\_note-robinson-casali2003-3](https://en.wikipedia.org/wiki/Intelligibility_(communication)#cite_note-robinson-casali2003-3). [Accessed: 04- Dec- 2021].

[9]D. Rodriguez, ece.uprm.edu, 2021. [Online]. Available:  
<http://ece.uprm.edu/~domingo/teaching/inel4301/Information/wav2asc.m>.

## F. Matlab Code

### Signal Generators:

```
1
2
3 - Fs=40000;
4 - Ts=1/Fs;
5 - %V=10;           %Time Transmission window (TW)
6 - tt=0:Ts:(2048*Ts)-Ts;    %Time axis for TW
7 - tz=0:Ts:(1024*Ts)-Ts;    %Pulse time tone "ON" width
8 - s1=cos(2*pi*697*tt)+cos(2*pi*1209*tt);
9 - s3=cos(2*pi*697*tt)+cos(2*pi*1477*tt);
10 - s8=cos(2*pi*852*tt)+cos(2*pi*1336*tt);
11 - s0=cos(2*pi*941*tt)+cos(2*pi*1336*tt);
12 - sz=0*tz;           %Pulse signal segment "OFF"
13 - s=[s1 sz s3 sz s8 sz s8 sz s3 sz s3 sz s1 sz s0 sz s0 sz s0 sz s1 sz];
14
15 - sound(s, Fs);
16
17 - audiowrite('dtmf01.wav', s, Fs);
```

**Code 1:** Matlab Script for dtmf signal

```
1 - Fs = 40000;
2 - Ts = 1/Fs;
3 - f1 = 4800;
4 - f2 = 5200;
5 - f3 = 5800;
6 - N = 30720;
7 - V = N*Ts;
8 - t = 0:Ts:V-Ts;
9
10 - % Signals
11
12 - g1 = cos(2*pi*f1*t);
13 - g2 = cos(2*pi*f2*t);
14 - g3 = cos(2*pi*f3*t);
15 - g=(1/80)*g1+(1/120)*g2+(1/240)*g3;
16
17 - audiowrite('uwtd01.wav', g, Fs);
18
```

**Code 2:** Matlab Script for uwtd signal

```

1 - clear all                                %Clear working space
2 - close all
3 - %%*****Input Signals*****
4 - %
5 - Fs=40000
6 - n = randn(30720,1);
7 - variance = sqrt(1/5000);
8 - n= variance* n;
9 - audiowrite('nois01.wav', n, Fs);
10 -

```

**Code 3:** Matlab Script for noise signal

### Tasks 1-5:

```

4 - [st, Fss] = audioread('S086GP01_EDP_INPls/dtmf01.wav');           % s(t) signal
5 - [gt, Fsg] = audioread('S086GP01_EDP_INPlg/uwtd01.wav');           % g(t) signal
6 - [nt,Fsn]=audioread('S086GP01_EDP_INPln/nois01.wav');             % n(t) signal
7 -
8 -
9 - %%****Processing Input Signals*****
10 -
11
12 - save st01.txt st -ascii                  %Save signal as s.txt file
13 - load -ascii st01.txt;                   %Load ascii file s.txt
14 - ssiz=length(st);                      %Get the length of the "s" signal
15
16 - save gt01.txt gt -ascii                  %Save signal as g.txt file
17 - load -ascii gt01.txt;                   %Load ascii file g.txt
18 - gsiz=length(gt);                      %Get the length of the "g" signal
19
20 - save nt01.txt nt -ascii                  %Save signal as n.txt file
21 - load -ascii nt01.txt;                   %Load ascii file n.txt
22 - nsiz=length(nt);                      %Get the length of the "n" signal
23
24 - %*****Parameter Settings*****
25 - %
26 - %
27 - Fs=40000;                            %Sampling rate
28 - Ts=1/Fs;
29 - fc=8000;                             %Carrier frequency
30 - N=ssiz;
31 - V=N*Ts;
32 - fincq=1/V;                          %Frequency resolution of wanted signal
33 - fkq=-((Fs+1)/2):fincq:+((Fs+1)/2)-fincq; %Frequency axis of wanted signal

```

```

32 - fincq=1/V;                                %Frequency resolution of wanted signal
33 - fkq=-((Fs+1)/2):fincq:+((Fs+1)/2)-fincq; %Frequency axis of wanted signal
34 -
35 - t=0:Ts:V-Ts;
36 - M=200;                                     %Order of the filters
37 - Fv=2000;
38 -
39 -
40 -%%
41 -%% TRANSMITTER SYSTEM
42 -%%
43 - c=cos(2*pi*fc*t);                         %Carrier signal
44 - xm=st+gt;
45 -
46 - audiowrite("S086GP01_EDP_INPlx/xmt01.wav",xm,Fs);
47 -

48 - cl=transpose(c);
49 - xc=xm.*cl;                                 %Modulated wanted signal plus interference
50 -
51 -%% Part #1 task #1
52 -%%
53 -%% BAND_PASS FILTER DESIGN (hB)
54 -%%
55 - fm=Fv;                                      %Cutoff frequency
56 - fUpper = fc + fm;
57 - fLower = fc - fm;
58 - Wnl = 2*(fLower)/Fs;
59 - Wnu = 2*(fUpper)/Fs;
60 - Fb = [Wnl Wnu];                            %band limits
61 - hB = fir1(M-1,Fb,'bandpass');            %filter design
62 - yci=conv(xc,hB,'same');                   %Removal of unwanted interference signals

64 - ycisiz=length(yci);
65 - Nyci=ycisiz;
66 - Vyci=Nyci*Ts;
67 - fincqyci=1/Vyci;                           %Frequency resolution of channel input signal
68 - fkqyci=-(Fs+1)/2:fincqyci:+((Fs+1)/2)-fincqyci; %Frequency axis of channel input signal
69 -
70 -
71 -%% Part #1 task #2
72 -%%
73 -%% LOW-PASS FILTER DESIGN (Tc)
74 -%%
75 - B = 12000;
76 - M=100;
77 - Wn=2*B/Fs;                                %Normilized frequency
78 - th=0:Ts:M*Ts-Ts;                          %Time axis to plot impulse response signal
79 - hc=fir1(M-1,Wn);                          %Impulse response signal
80 - yc=conv(yci,hc,'same');

```

```

82 -     ycsiz=length(yc);
83 -     Nyc=ycsiz;
84 -     Vyc=Nyc*Ts;
85 -     fincqyc=1/Vyc;           %Frequency resolution of channel filter output signal
86 -     fkqyc=-((Fs+1)/2):fincqyc:+((Fs+1)/2)-fincqyc; %Frequency axis of channel filter output signal
87 -
88 %% Part #1 task #3
89 %%
90 %% ADDITION OF NOISE SIGNAL (Tn)
91 -     yco = yc+nt;
92 -
93 -     ycosiz=length(yco);
94 -     Nyco=ycosiz;
95 -     Vyco=Nyco*Ts;
96 -     fincqyco=1/Vyco;          %Frequency resolution of channel output signal
97 -     fkqyco=-((Fs+1)/2):fincqyco:+((Fs+1)/2)-fincqyco; %Frequency axis of channel output signal
98 -
99 | %
100 -    xmSNR = snr(st,gt);
101 -    ycPdB = pow2db(bandpower(yc));
102 -    nPdB = pow2db(bandpower(nt));
103 -    SNR = snr(yc,nt);
104 -    nsig_avg = mean(nt)           %Mean of noise
105 -    nsig_var = var(nt)           %Variance of noise
106 -
107 %% Part #1 task #4
108 %%
109 %% BANDPASS FILTER (Tr)
110 %%
111 -    fR = [Wnl Wnu];             %band limits
112 -    hR = fir1(15,fR,'bandpass'); %filter design
113 -    ydi = conv(yco,hR,'same');   %Removal of unwanted interference signals
...
115 -    ydisiz = length(ydi);
116 -    Nydi = ydisiz;
117 -    Vydi = Nydi*Ts;
118 -    fincqydi = 1/Vydi;           %Frequency resolution of demodulator input signal
119 -    fkqydi = -((Fs+1)/2):fincqydi:+((Fs+1)/2)-fincqydi; %Frequency axis of demodulator input signal
120 -
121 -    ydiPdB = pow2db(bandpower(ydi));
122 -    SNR2 = snr(ydi-nt,nt);
123 -
124 %***** Demodulator Td*****
125 -    ydo = ydi.* (2*c1);
126 -
127 -    ydosiz=length(ydo);
128 -    Nydo=ydosiz;
129 -    Vydo=Nydo*Ts;
130 -    fincqydo=1/Vydo;            %Frequency resolution of demodulator output signal
131 -    fkqydo=-((Fs+1)/2):fincqydo:+((Fs+1)/2)-fincqydo; %Frequency axis of demodulator output signal
132

```

```

133 %% Part #1 Task #5
134 %%
135 %% LOW_PASS FILTER DESIGN (Tr)
136 %%
137 - Wn=2*fm/Fs; %Normalized cut-off frequency
138 - M=50; %Order of the impulse response signal hL(t)
139 - thL=0:Ts:M*Ts-Ts; %Time axis to plot impulse response signal
140 - hL=fir1(M-1,Wn); %Impulse response signal
141 - xr=conv(ydo,hL); %Convolution operation for filtering
142
143 audiowrite("S086GP01_EDP_OUP115/dtmf01.wav",xr,Fs);
144
145 Dr=((M-1)/2)*Ts
146 xrsiz=length(xr);
147 Nxr=xrsiz;
148 Vxr=Nxr*Ts;
149 fincqxr=1/Vxr; %Frequency resolution of received signal

150 fkqxr=-((Fs+1)/2):fincqxr:((Fs+1)/2)-fincqxr; %Frequency axis of received signal
151 %%
152 %%
153 %% *****PLOTS*****
154 %%
155 tyci=0:Ts:(length(yci)-1)*Ts;
156 tyc=0:Ts:(length(yc)-1)*Ts;
157 txr=0:Ts:(length(xr)-1)*Ts;
158
159
160 % % Time Domain
161
162 plot(t,st);
163 axis([0 (length(st)*Ts) min(st) max(st)]);
164 xlabel('Time in Seconds')

163 - axis([0 (length(st)*Ts) min(st) max(st)]);
164 - xlabel('Time in Seconds')
165 - ylabel('Amplitude')
166 - title('Wanted signal s(t)')
167 - grid
168
169 % %
170
171 figure
172 plot(t,gt);
173 axis([0 (length(gt)*Ts) min(gt) max(gt)]);
174 xlabel('Time in Seconds')
175 ylabel('Amplitude')
176 title('Interference Signal g(t)')
177 grid
178
179 % %
180

```

```

181 - figure
182 - plot(t,xm)
183 - xlabel('Time in Seconds')
184 - ylabel('Amplitude')
185 - title('Modulating Signal (xm(t) = s(t) + g(t))')
186 - grid
187
188 % %
189
190 - figure
191 - plot(t,xc)
192 - xlabel('Time in Seconds')
193 - ylabel('Amplitude')
194 - title('Modulated Signal xc(t)')
195 - grid
196
197 % %

197 % %
198
199 - figure
200 - plot(tyci,yci)
201 - axis([0, (length(yci)*Ts), min(yci), max(yci)]);
202 - xlabel('Time in Seconds')
203 - ylabel('Amplitude')
204 - title('Channel Input Signal yci(t)')
205 - grid
206
207 % %
208
209 - figure
210 - plot(tyci,yc)
211 - axis([0 (length(yc)*Ts) min(yc) max(yc)]);
212 - xlabel('Time in Seconds')
213 - ylabel('Amplitude')

218
219 - figure
220 - plot(tyc,nt)
221 - axis([0 (length(nt)*Ts) min(nt) max(nt)]);
222 - xlabel('Time in Seconds')
223 - ylabel('Amplitude')
224 - title('Channel Noise Signal n(t)')
225 - grid
226
227 % %
228
229 - figure
230 - plot(tyc,yco)
231 - axis([0 (length(yco)*Ts) min(yco) max(yco)]);
232 - xlabel('Time in Seconds')
233 - ylabel('Amplitude')
234 - title('Channel Output Signal yco(t)')
235 - grid

```

```

239 - figure
240 - plot(tyc,ydi)
241 - axis([0 (length(ydi)*Ts) min(ydi) max(ydi)]);
242 - xlabel('Time in Seconds')
243 - ylabel('Amplitude')
244 - title('Demodulator Input Signal ydi(t)')
245 - grid
246
247 % %
248
249 - figure
250 - plot(tyc,ydo)
251 - %axis([0, (length(ydo)*Ts), min(ydo), max(ydo)]); ****
252 - xlabel('Time in Seconds')
253 - ylabel('Amplitude')
254 - title('Demodulator Output Signal ydo(t)')
255 - grid
256
259 - figure
260 - plot(txr,xr)
261 - %axis([0 (length(xr)*Ts) min(xr) max(xr)]);
262 - xlabel('Time in Seconds')
263 - ylabel('Amplitude')
264 - title('Received Signal xr(t)')
265 - grid
266
267 % % Spectrums
268
269 - fw=fft(st); %Spectrum of wanted signal (Fund. Reg.)
270 - sfs=fftshift(fw); %Spectrum of wanted signal (Princ. Reg.)
271 - asfs=abs(sfs); %Magnitude of spectrum of wanted signal
272
273 %
274
275 - fg=fft(gt); %Spectrum of interference signal (Fund. Reg.)
276 - sfg=fftshift(fg); %Spectrum of interference signal (Princ. Reg.)
277 - asig=abs(sfg); %Magnitude of spectrum of wanted signal
278
279 %
280
281 - fxm=fft(xm); %Spectrum of input signal (Fund. Reg.)
282 - sfxm=fftshift(fxm); %Spectrum of input signal (Princ. Reg.)
283 - asfxm=abs(sfxm); %Magnitude of spectrum of input signal
284
285 %
286
287 - fxc=fft(xc); %Spectrum of modulated signal (Fund. Reg.)
288 - sfxc=fftshift(fxc); %Spectrum of modulated signal (Princ. Reg.)
289 - asfxc=abs(sfxc); %Magnitude of spectrum of modulated signal
290
291 %
292
293 - fyci=fft(yci); %Spectrum of channel input signal (Fund. Reg.)
294 - sfyci=fftshift(fyci); %Spectrum of channel input signal (Princ. Reg.)

```

```

295 - asfyci=abs(sfyci); %Magnitude of spectrum of channel input signal
296
297 %
298
299 - fyc=fft(yc); %Spectrum of channel filter output signal (Fund. Reg.)
300 - sfyc=fftshift(fyc); %Spectrum of channel filter output signal (Princ. Reg.)
301 - asfyc=abs(sfyc); %Magnitude of spectrum of channel filter output signal
302
303 %
304
305 - fn=fft(nt); %Spectrum of AWGN signal (Fund. Reg.)
306 - sfn=fftshift(fn); %Spectrum of AWGN signal (Princ. Reg.)
307 - asfn=abs(sfn); %Magnitude of spectrum of wanted signal
308
309 %
310
311 - fyco=fft(yco); %Spectrum of channel output signal (Fund. Reg.)
312 - sfyco=fftshift(fyco); %Spectrum of channel output signal (Princ. Reg.)

313 - asfyco=abs(sfyci); %Magnitude of spectrum of channel output signal
314
315 %
316
317 - fydi=fft(ydi); %Spectrum of demodulator input signal (Fund. Reg.)
318 - sfydi=fftshift(fydi); %Spectrum of demodulator input signal (Princ. Reg.)
319 - asfydi=abs(sfyci); %Magnitude of spectrum of demodulator input signal
320
321 %
322
323 - fydo=fft(ydo); %Spectrum of demodulator output signal (Fund. Reg.)
324 - sfydo=fftshift(fydo); %Spectrum of demodulator output signal (Princ. Reg.)
325 - asfydo=abs(sfyci); %Magnitude of spectrum of demodulator output signal
326
327 %
328
329 - fxr=fft(xr); %Spectrum of received signal (Fund. Reg.)
330 - sfxr=fftshift(fxr); %Spectrum of received signal (Princ. Reg.)

331 - asfxr=abs(sfyci); %Magnitude of spectrum of received signal
332
333 %
334
335 - figure
336 - plot(fkq,asfs)
337 - xlabel('Frequency in Hertz')
338 - ylabel('Magnitude')
339 - title('Magnitude of Spectrum of Wanted Signal S(f)')
340 - grid
341
342 %
343
344 - figure
345 - plot(fkq,asfg)
346 - xlabel('Frequency in Hertz')
347 - ylabel('Magnitude')

```

```

348 - title('Magnitude of Spectrum of Interference Signal G(f)')
349 - grid
350
351 % %
352
353 - figure
354 - plot(fkq,asfxm)
355 - xlabel('Freq in Hertz')
356 - ylabel('Magnitude')
357 - title('Magnitude of Spectrum of Modulating Signal Xm(f)')
358 - grid
359
360 % %
361
362 - figure
363 - plot(fkq,asfxc)
364 - xlabel('Freq in Hertz')
365 - ylabel('Magnitude')

366 - title('Magnitude of Spectrum of Modulated Signal Xc(f)')
367 - grid
368
369 % %
370
371 - asfyci=transpose(asfyci);
372 - figure
373 - plot(fkqy,asfyci)
374 - xlabel('Frequency in Hertz')
375 - ylabel('Magnitude')
376 - title('Magnitude of Spectrum of Channel Input Signal Yci(f)')
377 - grid
378
379 % %
380
381 - asfyc=transpose(asfyc);
382 - figure
383 - plot(fkqy,asfyc)

384 - xlabel('Frequency in Hertz')
385 - ylabel('Magnitude')
386 - title('Magnitude of Spectrum of Channel Filter Output Signal Yc(f)')
387 - grid
388
389 % %
390
391 - figure
392 - plot(fkq,asfn)
393 - xlabel('Frequency in Hertz')
394 - ylabel('Magnitude')
395 - title('Magnitude of Spectrum of AWGN Signal N(f)')
396 - grid
397
398 % %
399
400 - asfyco=transpose(asfyco);

```

```

400 - asfyco=transpose(asfyco);
401 - figure
402 - plot(fkqyco,asfyco)
403 - xlabel('Frequency in Hertz')
404 - ylabel('Magnitude')
405 - title('Magnitude of Spectrum of Channel Output Signal Yco(f)')
406 - grid
407
408 % %
409
410 - asfydi=transpose(asfydi);
411 - figure
412 - plot(fkqydi,asfydi)
413 - xlabel('Frequency in Hertz')
414 - ylabel('Magnitude')
415 - title('Magnitude of Spectrum of Demodulator Input Signal Ydi(f)')
416 - grid
417
420 - figure
421 - plot(fkqydo,asfydo)
422 - xlabel('Frequency in Hertz')
423 - ylabel('Magnitude')
424 - title('Magnitude of Spectrum of Demodulator Output Signal Ydo(f)')
425 - grid
426
427 % %
428
429 - figure
430 - plot(fkqxr,asfxr)
431 - xlabel('Frequency in Hertz')
432 - ylabel('Magnitude')
433 - title('Magnitude of Spectrum of Received Signal Xr(f)')
434 - grid
435
436 % %
437
438 - sound(st, Fs);
439 - pause(3);
440 - sound(xr, Fs);

```

**Code 4: Matlab Script for Tasks 1-5**

## Task 6:

```
1 %*****Wanted Signal Input/Output*****
2 %*****Wanted Signal Input/Output*****
3 %
4 - [st, Fss] = audioread('S086GP01_EDP_INP1s/dtmf01.wav');      % s(t) signal
5 - [gt, Fsg] = audioread('S086GP01_EDP_INP1g/uwtd01.wav');      % g(t) signal
6 - [nt, Fsn]=audioread('S086GP01_EDP_INP1n/nois01.wav');        % n(t) signal
7
8
9 %
10 %%****Processing Input Signals****
11 %
12
13 - save st01.txt st -ascii                         %Save signal as s.txt file
14 - load -ascii st01.txt;                          %Load ascii file s.txt
15 - ssiz=length(st);                            %Get the length of the "s" signal
16
17 - save gt01.txt gt -ascii                         %Save signal as g.txt file
18 - load -ascii gt01.txt;                          %Load ascii file g.txt
19 - gsiz=length(gt);                            %Get the length of the "g" signal
20
21 - save nt01.txt nt -ascii                         %Save signal as n.txt file
22 - load -ascii nt01.txt;                          %Load ascii file n.txt
23 - nsiz=length(nt);                            %Get the length of the "n" signal
24
25 %*****Parameter Settings*****
26 %
27 %
28 - Fs=40000;                                     %Sampling rate
29 - Ts=1/Fs;
30 - fc=8000;                                      %Carrier frequency
31 - N=ssiz;
32 - V=N*Ts;
33 - fincq=1/V;                                     %Frequency resolution of wanted signal
34 - fkq=-( (Fs+1)/2 ):fincq:+((Fs+1)/2)-fincq; %Frequency axis of wanted signal
35
36 - t=0:Ts:V-Ts;
37 - M=100;                                         %Order of the filters
38 - Fv=2000;
```

```

41 %%  

42 %% TRANSMITTER SYSTEM  

43 %%  

44 - c=cos(2*pi*fc*t); %Carrier signal  

45 - xm=st+gt;  

46 - cl=transpose(c);  

47 - xc=xm.*cl; %Modulated wanted signal plus interference  

48  

49 %% Part #1 task #1  

50 %%  

51 %% BAND_PASS FILTER DESIGN (hB)  

52 %%  

53 - fm=Fv; %Cutoff frequency  

54 - fUpper = fc + fm;  

55 - fLower = fc - fm;  

56 - Wnl = 2*(fLower)/Fs;  

57 - Wnl2 = 2*(fc)/Fs;  

58 - Wnu = 2*(fUpper)/Fs;  

59 - Fb = [Wnl2 Wnu]; %band limits  

60 - hB = fir1(M-1,Fb,'bandpass'); %filter design  

61 - yci=conv(xc,hB,'same'); %Removal of unwanted interference signals  

62  

63 - ycisz=length(yci);  

64 - Nyci=ycisz;  

65 - Vyci=Nyci*Ts;  

66 - fincqyci=1/Vyci; %Frequency resolution of channel input signal  

67 - fkqyci=-(Fs+1)/2:fincqyci:+((Fs+1)/2)-fincqyci; %Frequency axis of channel input signal  

68  

69 %% Part #1 task #2  

70 %%  

71 %% LOW-PASS FILTER DESIGN (Tc)  

72 %%  

73 - B = 12000; %Normilized frequency  

74 - Wn=2*B/Fs; %Time axis to plot impulse response signal  

75 - th=0:Ts:M*Ts-Ts; %Impulse response signal  

76 - hc=fir1(M-1,Wn);  

77 - yc=conv(yci,hc,'same');  

78  

79 - ycsiz=length(yc);  

80 - Nycc=ycsiz;  

81 - Vyc=Nycc*Ts;  

82 - fincqyc=1/Vyc; %Frequency resolution of channel filter output signal  

83 - fkqyc=-(Fs+1)/2:fincqyc:+((Fs+1)/2)-fincqyc; %Frequency axis of channel filter output signal  

84 -  

85 %% Part #1 task #3  

86 %%  

87 %% ADDITION OF NOISE SIGNAL (Tn)  

88 - yco = yc+nt;  

89  

90 - ycosiz=length(yco);  

91 - Nyco=ycosiz;  

92 - Vyco=Nyco*Ts;  

93 - fincqyco=1/Vyco; %Frequency resolution of channel output signal  

94 - fkqyco=-(Fs+1)/2:fincqyco:+((Fs+1)/2)-fincqyco; %Frequency axis of channel output signal  

95 -  

96 - xmSNR = snr(st,gt);  

97 - ycPdB = pow2db(bandpower(yc));  

98 - nPdB = pow2db(bandpower(nt));  

99 - SNR = snr(yc,nt);  

100 -  

101 - nsig_avg = mean(nt) %Mean of noise  

102 - nsig_var = var(nt) %Variance of noise  

103 -  

104 -

```

```

105 %% Part #1 task #4
106 %%
107 %% BANDPASS FILTER (Tr)
108 %%
109 - fR = [Wnl Wnu]; %band limits
110 - hR = fir1(15,fR,'bandpass'); %filter design
111 - ydi = conv(yco,hR,'same'); %Removal of unwanted interference signals
112
113 - ydisiz = length(ydi);
114 - Nydi = ydisiz;
115 - Vydi = Nydi*Ts;
116 - fincqydi = 1/Vydi; %Frequency resolution of demodulator input signal
117 - fkqydi = -( (Fs+1)/2 ):fincqydi:+((Fs+1)/2)-fincqydi; %Frequency axis of demodulator input signal
118
119 - ydiPdB = pow2db(bandpower(ydi));
120 - SNR2 = snr(ydi,nt);
121
122 %***** Demodulator Td*****
123 - ydo = ydi.* (2*c1);
124
125 - ydosiz=length(ydo);
126 - Nydo=ydosiz;
127 - Vydo=Nydo*Ts;
128
129 - Vydo=Nydo*Ts;
130 - fincqydo=1/Vydo; %Frequency resolution of demodulator output signal
131 - fkqydo=-( (Fs+1)/2 ):fincqydo:+((Fs+1)/2)-fincqydo; %Frequency axis of demodulator output signal
132
133 %% Part #1 Task #5
134 %%
135 %% LOW_PASS FILTER DESIGN (Tr)
136 %%
137 - Wn=2*f0/Fs; %Normalized cut-off frequency
138 - M=50; %Order of the impulse response signal hL(t)
139 - thL=0:Ts:M*Ts-Ts; %Time axis to plot impulse response signal
140 - hL=fir1(M-1,Wn); %Impulse response signal
141 - xr=conv(ydo,hL); %Convolution operation for filtering
142
143 - audiowrite("S086GP01_EDP_OUP166/dtmf01.wav",xr,Fs);
144
145 - Dr=((M-1)/2)*Ts;
146 - xrsiz=length(xr);
147 - Nxr=xrsiz;
148 - Vxr=Nxr*Ts;
149 - fincqxr=1/Vxr; %Frequency resolution of received signal
150 - fkqxr=-( (Fs+1)/2 ):fincqxr:+((Fs+1)/2)-fincqxr; %Frequency axis of received signal
151 %% *****PLOTS*****
152
153 %% Time Domain
154 %% PLOTS
155 %% Time Domain
156 %% PLOTS
157 %% Time Domain
158 %% PLOTS
159 %% Time Domain
160 %% PLOTS
161 %% Time Domain
162 %% PLOTS
163 %% Time Domain
164 %% PLOTS
165 %% Time Domain
166 %% PLOTS
167 %% Time Domain
168 %% PLOTS
169 %% Time Domain
170 %% PLOTS
171 %% Time Domain
172 %% PLOTS
173 %% Time Domain
174 %% PLOTS

```

```

174 - title('Interference Signal g(t)')
175 - grid
176
177 % %
178
179 - figure
180 - plot(t,xm)
181 - xlabel('Time in Seconds')
182 - ylabel('Amplitude')
183 - title('Modulating Signal (xm(t) = s(t) + g(t))')
184 - grid
185
186 % %
187
188 - figure
189 - plot(t,xc)
190 - xlabel('Time in Seconds')
191 - ylabel('Amplitude')
192 - title('Modulated Signal xc(t)')
193 - grid
194
195 % %
196
197 - figure
198 - plot(tyci,yci)
199 - axis([0, (length(yci)*Ts), min(yci), max(yci)]);
200 - xlabel('Time in Seconds')
201 - ylabel('Amplitude')
202 - title('Channel Input Signal yci(t)')
203 - grid

```

```

206
207 - figure
208 - plot(tyci,yc)
209 - axis([0 (length(yc)*Ts) min(yc) max(yc)]);
210 - xlabel('Time in Seconds')
211 - ylabel('Amplitude')
212 - title('Channel Filter Output Signal yc(t)')
213 - grid
214
215 % %
216
217 - figure
218 - plot(tyc,nt)
219 - axis([0 (length(nt)*Ts) min(nt) max(nt)]);
220 - xlabel('Time in Seconds')
221 - ylabel('Amplitude')
222 - title('Channel Noise Signal n(t)')
223 - grid
224
225 % %
226
227 - figure
228 - plot(tyc,yco)
229 - axis([0 (length(yco)*Ts) min(yco) max(yco)]);
230 - xlabel('Time in Seconds')
231 - ylabel('Amplitude')
232 - title('Channel Output Signal yco(t)')
233 - grid
234
235 % %

```

```

236
237 -    figure
238 -    plot(tyc,ydi)
239 -    axis([0 (length(ydi)*Ts) min(ydi) max(ydi)]);
240 -    xlabel('Time in Seconds')
241 -    ylabel('Amplitude')
242 -    title('Demodulator Input Signal ydi(t)')
243 -    grid
244
245 % %
246
247 -    figure
248 -    plot(tyc,ydo)
249 -    %axis([0, (length(ydo)*Ts), min(ydo), max(ydo)]); ****
250 -    xlabel('Time in Seconds')
251 -    ylabel('Amplitude')
252 -    title('Demodulator Output Signal ydo(t)')
253 -    grid
254
255 % %
256
257 -    figure
258 -    plot(txr,xr)
259 -    %axis([0 (length(xr)*Ts) min(xr) max(xr)]);
260 -    xlabel('Time in Seconds')
261 -    ylabel('Amplitude')
262 -    title('Received Signal xr(t)')
263 -    grid
264
265 % % Spectrums

```

```

264
265 % % Spectrums
266
267 - fw=fft(st); %Spectrum of wanted signal (Fund. Reg.)
268 - sfs=fftshift(fw); %Spectrum of wanted signal (Princ. Reg.)
269 - asfs=abs(sfs); %Magnitude of spectrum of wanted signal
270
271 %
272
273 - fg=fft(gt); %Spectrum of interference signal (Fund. Reg.)
274 - sfg=fftshift(fg); %Spectrum of interference signal (Princ. Reg.)
275 - asfg=abs(sfg); %Magnitude of spectrum of wanted signal
276
277 %
278
279 - fxm=fft(xm); %Spectrum of input signal (Fund. Reg.)
280 - sfxm=fftshift(fxm); %Spectrum of input signal (Princ. Reg.)
281 - asfxm=abs(sfxm); %Magnitude of spectrum of input signal
282
283 %
284
285 - fxc=fft(xc); %Spectrum of modulated signal (Fund. Reg.)
286 - sfxc=fftshift(fxc); %Spectrum of modulated signal (Princ. Reg.)
287 - asfxc=abs(sfxc); %Magnitude of spectrum of modulated signal
288
289 %
290
291 - fyci=fft(yci); %Spectrum of channel input signal (Fund. Reg.)
292 - sfyci=fftshift(fyci); %Spectrum of channel input signal (Princ. Reg.)
293 - asfyci=abs(sfyci); %Magnitude of spectrum of channel input signal
294 %
295
296
297 - fyc=fft(yc); %Spectrum of channel filter output signal (Fund. Reg.)
298 - sfyc=fftshift(fyc); %Spectrum of channel filter output signal (Princ. Reg.)
299 - asfyc=abs(sfyc); %Magnitude of spectrum of channel filter output signal
300
301 %
302
303 - fn=fft(nt); %Spectrum of AWGN signal (Fund. Reg.)
304 - sfn=fftshift(fn); %Spectrum of AWGN signal (Princ. Reg.)
305 - asfn=abs(sfn); %Magnitude of spectrum of wanted signal
306
307 %
308
309 - fyco=fft(yco); %Spectrum of channel output signal (Fund. Reg.)
310 - sfyco=fftshift(fyco); %Spectrum of channel output signal (Princ. Reg.)
311 - asfyco=abs(sfyc); %Magnitude of spectrum of channel output signal
312
313 %
314
315 - fydi=fft(ydi); %Spectrum of demodulator input signal (Fund. Reg.)
316 - sfydi=fftshift(fydi); %Spectrum of demodulator input signal (Princ. Reg.)
317 - asfydi=abs(sfydi); %Magnitude of spectrum of demodulator input signal
318
319 %
320
321 - fydo=fft(ydo); %Spectrum of demodulator output signal (Fund. Reg.)
322 - sfydo=fftshift(fydo); %Spectrum of demodulator output signal (Princ. Reg.)

```

```

323 - asfydo=abs(sfyo); %Magnitude of spectrum of demodulator output signal
324 -
325 %
326
327 - fxr=fft(xr); %Spectrum of received signal (Fund. Reg.)
328 - sfxr=fftshift(fxr); %Spectrum of received signal (Princ. Reg.)
329 - asfxr=abs(sfxr); %Magnitude of spectrum of received signal
330
331 %
332
333 - figure
334 - plot(fkq,asfs)
335 - xlabel('Frequency in Hertz')
336 - ylabel('Magnitude')
337 - title('Magnitude of Spectrum of Wanted Signal S(f)')
338 - grid
339 %
340 %
341
342 - figure
343 - plot(fkq,asfg)
344 - xlabel('Frequency in Hertz')
345 - ylabel('Magnitude')
346 - title('Magnitude of Spectrum of Interference Signal G(f)')
347 - grid
348 %
349 %
350
351 - figure
352 - plot(fkq,asfxm)

353 - xlabel('Freq in Hertz')
354 - ylabel('Magnitude')
355 - title('Magnitude of Spectrum of Modulating Signal Xm(f)')
356 - grid
357
358 %
359
360 - figure
361 - plot(fkq,asfxc)
362 - xlabel('Freq in Hertz')
363 - ylabel('Magnitude')
364 - title('Magnitude of Spectrum of Modulated Signal Xc(f)')
365 - grid
366
367 %
368
369 - asfyci=transpose(asfyci);
370 - figure
371 - plot(fkqy,asfyi)
372 - xlabel('Frequency in Hertz')
373 - ylabel('Magnitude')
374 - title('Magnitude of Spectrum of Channel Input Signal Yci(f)')
375 - grid
376
377 %
378
379 - asfyc=transpose(asfyc);
380 - figure
381 - plot(fkqy,asfyc)
382 - xlabel('Frequency in Hertz')

```

```

383 - ylabel('Magnitude')
384 - title('Magnitude of Spectrum of Channel Filter Output Signal Yc(f)')
385 - grid
386
387 % %
388
389 - figure
390 - plot(fkq,asfn)
391 - xlabel('Frequency in Hertz')
392 - ylabel('Magnitude')
393 - title('Magnitude of Spectrum of AWGN Signal N(f)')
394 - grid
395
396 % %
397
398 - asfyco=transpose(asfyco);
399 - figure
400 - plot(fkqyco,asfyco)
401 - xlabel('Frequency in Hertz')
402 - ylabel('Magnitude')
403 - title('Magnitude of Spectrum of Channel Output Signal Yco(f)')
404 - grid
405
406 % %
407
408 - asfydi=transpose(asfydi);
409 - figure
410 - plot(fkqydi,asfydi)
411 - xlabel('Frequency in Hertz')
412 - ylabel('Magnitude')

413 - title('Magnitude of Spectrum of Demodulator Input Signal Ydi(f)')
414 - grid
415
416 % %
417
418 - figure
419 - plot(fkqydo,asfydo)
420 - xlabel('Frequency in Hertz')
421 - ylabel('Magnitude')
422 - title('Magnitude of Spectrum of Demodulator Output Signal Ydo(f)')
423 - grid
424
425 % %
426
427 - figure
428 - plot(fkqxr,asfxr)
429 - xlabel('Frequency in Hertz')
430 - ylabel('Magnitude')
431 - title('Magnitude of Spectrum of Received Signal Xr(f)')
432 - grid
433
434 % %
435
436 - sound(st, Fs);
437 - pause(3);
438 - sound(xr, Fs);

```

**Code 5: Matlab Script for Tasks 6**

## Task 2-1:

```

1 %*****
2 %*****Wanted Signal Input/Output*****
3 %*****
4 [st, Fss] = audioread('S086GP01_EDP_INP1s/dtmf01.wav'); % s(t) signal
5 [gt, Fsg] = audioread('S086GP01_EDP_INP1g/uwtd01.wav'); % g(t) signal
6 [nt,Fsn]=audioread('S086GP01_EDP_INP1n/nois01.wav'); % n(t) signal
7 %
8 %
9 %%Processing Input Signals****
10 %
11
12 save st01.txt st -ascii %Save signal as s.txt file
13 load -ascii st01.txt; %Load ascii file s.txt
14 ssiz=length(st); %Get the length of the "s" signal
15
16 save gt01.txt gt -ascii %Save signal as g.txt file
17 load -ascii gt01.txt; %Load ascii file g.txt
18 gsiz=length(gt); %Get the length of the "g" signal
19
20 save nt01.txt nt -ascii %Save signal as n.txt file
21 load -ascii nt01.txt; %Load ascii file n.txt
22 nsiz=length(nt); %Get the length of the "n" signal
23
24 %*****
25 %*****Parameter Settings*****
26 %*****
27 Fs=40000; %Sampling rate
28 Ts=1/Fs;
29 fc=8000; %Carrier frequency
30 N=ssiz;

31 V=N*Ts;
32 fincq=1/V; %Frequency resolution of wanted signal
33 fkq=-( (Fs+1)/2 ):fincq:+((Fs+1)/2)-fincq; %Frequency axis of wanted signal
34
35 t=0:Ts:V-Ts;
36 M=100; %Order of the filters
37 Fv=2000;
38
39
40 %%
41 %% TRANSMITTER SYSTEM
42 %%
43 c=cos(2*pi*fc*t); %Carrier signal
44 xm=st+gt;
45
46 %QUANTIZER
47 n=3; %Number of bits
48 maxxm = max(xm);
49 i = 2*maxxm/(2^n -1); %Interval length
50 mxxm = maxxm + i; %One interval above max value (to have intervals of [-max,max])
51 partition = [-maxxm:i:maxxm]; %Quantization intervals
52 codebook = [-maxxm:i:mxxm]; %Value quantizer assigns
53 [index, quants] = quantiz(xm, partition, codebook);
54 xm = transpose(quants);
55 audiowrite("S086GP01_EDP_INP2qa/xmt01.wav",xm,Fs);
56 %
57
58 cl=transpose(c);
59 xc=xm.*cl; %Modulated wanted signal plus interference
60

```

```

61 %% Part #1 task #1
62 %%
63 %% BAND_PASS FILTER DESIGN (hB)
64 %%
65 fm=Fv; %Cutoff frequency
66 fUpper = fc + fm;
67 fLower = fc - fm;
68 Wnl = 2*(fLower)/Fs;
69 Wnu = 2*(fUpper)/Fs;
70 Fb = [Wnl Wnu]; %band limits
71 hB = fir1(M-1,Fb,'bandpass'); %filter design
72 yci=conv(xc,hB,'same'); %Removal of unwanted interference signals
73
74 ycisiz=length(yci);
75 Nyci=ycisiz;
76 Vyci=Nyci*Ts;
77 fincqyci=1/Vyci; %Frequency resolution of channel input signal
78 fkqyci=-(Fs+1)/2:fincqyci:((Fs+1)/2)-fincqyci; %Frequency axis of channel input signal
79
80
81 %% Part #1 task #2
82 %%
83 %% LOW-PASS FILTER DESIGN (Tc)
84 %%
85 B = 12000;
86 M=100;
87 Wn=2*B/Fs; %Normalized frequency
88 th=0:Ts:M*Ts-Ts; %Time axis to plot impulse response signal
89 hc=fir1(M-1,Wn); %Impulse response signal
90 yc=conv(yci,hc,'same');

91 ycsiz=length(yc);
92 Nyco=ycsiz;
93 Vyc=Vyc*Ts;
94 fincqyc=1/Vyc; %Frequency resolution of channel filter output signal
95 fkqyc=-(Fs+1)/2:fincqyc:((Fs+1)/2)-fincqyc; %Frequency axis of channel filter output signal
96
97 %% Part #1 task #3
98 %%
99 %% ADDITION OF NOISE SIGNAL (Tn)
100 yco = yc+nt;
101
102 ycosiz=length(yco);
103 Nyco=ycosiz;
104 Vyc=Vyc*Ts;
105 fincqyco=1/Vyc; %Frequency resolution of channel output signal
106 fkqyco=-(Fs+1)/2:fincqyco:((Fs+1)/2)-fincqyco; %Frequency axis of channel output signal
107
108 xmSNR = snr(st,gt);
109 ycpdB = pow2db(bandpower(yc));
110 npdB = pow2db(bandpower(nt));
111 SNR = snr(yc,nt);
112
113 nsig_avg = mean(nt); %Mean of noise
114 nsig_var = var(nt); %Variance of noise
115
116 %% Part #1 task #4
117 %%
118 %% BANDPASS FILTER (Tr)
119 %%
120

```

```

121 - fR = [Wnl Wnu];                                %band limits
122 - hR = fir1(15,fR,'bandpass');                 %filter design
123 - ydi = conv(yco,hR,'same');                   %Removal of unwanted interference signals
124
125 - ydisiz = length(ydi);
126 - Nydi = ydisiz;
127 - Vydi = Nydi*Ts;
128 - fincqydi = 1/Vydi;                            %Frequency resolution of demodulator input signal
129 - fkqydi = -( (Fs+1)/2 ) : fincqydi : +((Fs+1)/2) - fincqydi; %Frequency axis of demodulator input signal
130
131 - ydiPdB = pow2db(bandpower(ydi));
132 - SNR2 = snr(ydi-nt,nt)
133
134 - %***** Demodulator Td*****
135 - ydo = ydi.* (2*c1);
136
137 - ydosiz=length(ydo);
138 - Nydo=ydosiz;
139 - Vydo=Nydo*Ts;
140 - fincqydo=1/Vydo;                             %Frequency resolution of demodulator output signal
141 - fkqydo=-( (Fs+1)/2 ) : fincqydo : +((Fs+1)/2) - fincqydo; %Frequency axis of demodulator output signal
142
143 - %% Part #1 Task #5
144 - %%
145 - %% LOW_PASS FILTER DESIGN (Tr)
146 - %%
147 - Wn=2*fm/Fs;                                %Normalized cut-off frequency
148 - M=50;                                       %Order of the impulse response signal hL(t)
149 - thL=0:Ts:M*Ts-Ts;   %Time axis to plot impulse response signal
150 - hL=fir1(M-1,Wn);   %Impulse response signal
151 - xr=conv(ydo,hL);                           %Convolution operation for filtering
152
153 - audiowrite("S086GP01_EDP_OUP201/dtmf01.wav",xr,Fs);
154
155 - Dr=((M-1)/2)*Ts
156 - xrsiz=length(xr);
157 - Nxr=xrsiz;
158 - Vxr=Nxr*Ts;
159 - fincqxr=1/Vxr;                            %Frequency resolution of received signal
160 - fkqxr=-( (Fs+1)/2 ) : fincqxr : +((Fs+1)/2) - fincqxr; %Frequency axis of received signal
161
162 - %%
163 - %% *****PLOTS*****
164 - %%
165 - tyci=0:Ts:(length(yci)-1)*Ts;
166 - tyc=0:Ts:(length(yc)-1)*Ts;
167 - txr=0:Ts:(length(xr)-1)*Ts;
168
169
170 - % % Time Domain
171
172 - plot(t,st);
173 - axis([0 (length(st))*Ts min(st) max(st)]);
174 - xlabel('Time in Seconds')
175 - ylabel('Amplitude')
176 - title('Wanted signal s(t)')
177 - grid
178
179 - % %
180

```

```

181 - figure
182 - plot(t,gt);
183 - axis([0 (length(gt)*Ts) min(gt) max(gt)]);
184 - xlabel('Time in Seconds')
185 - ylabel('Amplitude')
186 - title('Interference Signal g(t)')
187 - grid
188
189 % %
190
191 - figure
192 - plot(t,xm)
193 - xlabel('Time in Seconds')
194 - ylabel('Amplitude')
195 - title('Modulating Signal (xm(t) = s(t) + g(t))')
196 - grid
197
198 % %
199
200 - figure
201 - plot(t,xc)
202 - xlabel('Time in Seconds')
203 - ylabel('Amplitude')
204 - title('Modulated Signal xc(t)')
205 - grid
206
207 % %
208
209 - figure
210 - plot(tyci,yci)

```

```
211 - axis([0, (length(yci)*Ts), min(yci), max(yci)]);
212 - xlabel('Time in Seconds')
213 - ylabel('Amplitude')
214 - title('Channel Input Signal yci(t)')
215 - grid
216
217 % %
218
219 - figure
220 - plot(tyci,yc)
221 - axis([0 (length(yc)*Ts) min(yc) max(yc)]);
222 - xlabel('Time in Seconds')
223 - ylabel('Amplitude')
224 - title('Channel Filter Output Signal yc(t)')
225 - grid
226
227 % %
228
229 - figure
230 - plot(tyc,nt)
231 - axis([0 (length(nt)*Ts) min(nt) max(nt)]);
232 - xlabel('Time in Seconds')
233 - ylabel('Amplitude')
234 - title('Channel Noise Signal n(t)')
235 - grid
236
237 % %
238
239 - figure
240 - plot(tyc,yco)
```

---

```

241 - axis([0 (length(yco)*Ts) min(yco) max(yco)]);
242 - xlabel('Time in Seconds')
243 - ylabel('Amplitude')
244 - title('Channel Output Signal yco(t)')
245 - grid
246
247 % %
248
249 figure
250 plot(tyc,ydi)
251 axis([0 (length(ydi)*Ts) min(ydi) max(ydi)]);
252 xlabel('Time in Seconds')
253 ylabel('Amplitude')
254 title('Demodulator Input Signal ydi(t)')
255 grid
256
257 % %
258
259 figure
260 plot(tyc,ydo)
261 %axis([0, (length(ydo)*Ts), min(ydo), max(ydo)]); ****
262 xlabel('Time in Seconds')
263 ylabel('Amplitude')
264 title('Demodulator Output Signal ydo(t)')
265 grid
266
267 % %
268
269 figure
270 plot(txr,xr)

```

```

271 %axis([0 (length(xr)*Ts) min(xr) max(xr)]);
272 xlabel('Time in Seconds')
273 ylabel('Amplitude')
274 title('Received Signal xr(t)')
275 grid
276
277 % % Spectrums
278
279 fw=fft(st); %Spectrum of wanted signal (Fund. Reg.)
280 sfs=fftshift(fw); %Spectrum of wanted signal (Princ. Reg.)
281 asfs=abs(sfs); %Magnitude of spectrum of wanted signal
282
283 %
284
285 fg=fft(gt); %Spectrum of interference signal (Fund. Reg.)
286 sfg=fftshift(fg); %Spectrum of interference signal (Princ. Reg.)
287 asfg=abs(sfg); %Magnitude of spectrum of wanted signal
288
289 %
290
291 fxm=fft(xm); %Spectrum of input signal (Fund. Reg.)
292 sfxm=fftshift(fxm); %Spectrum of input signal (Princ. Reg.)
293 asfxm=abs(sfxm); %Magnitude of spectrum of input signal
294
295 %
296
297 fxc=fft(xc); %Spectrum of modulated signal (Fund. Reg.)
298 sfxc=fftshift(fxc); %Spectrum of modulated signal (Princ. Reg.)
299 asfxc=abs(sfxc); %Magnitude of spectrum of modulated signal
300

```

```

301 %
302
303 - fyci=fft(yci); %Spectrum of channel input signal (Fund. Reg.)
304 - sfyci=fftshift(sfyci); %Spectrum of channel input signal (Princ. Reg.)
305 - asfyci=abs(sfyci); %Magnitude of spectrum of channel input signal
306
307 %
308
309 - fyc=fft(yc); %Spectrum of channel filter output signal (Fund. Reg.)
310 - sfyc=fftshift(sfyc); %Spectrum of channel filter output signal (Princ. Reg.)
311 - asfyc=abs(sfyc); %Magnitude of spectrum of channel filter output signal
312
313 %
314
315 - fn=fft(nt); %Spectrum of AWGN signal (Fund. Reg.)
316 - sfn=fftshift(sfn); %Spectrum of AWGN signal (Princ. Reg.)
317 - asfn=abs(sfny); %Magnitude of spectrum of wanted signal
318
319 %
320
321 - fyco=fft(yco); %Spectrum of channel output signal (Fund. Reg.)
322 - sfyco=fftshift(sfyc); %Spectrum of channel output signal (Princ. Reg.)
323 - asfyco=abs(sfyc); %Magnitude of spectrum of channel output signal
324
325 %
326
327 - fydi=fft(ydi); %Spectrum of demodulator input signal (Fund. Reg.)
328 - sfydi=fftshift(sfydi); %Spectrum of demodulator input signal (Princ. Reg.)
329 - asfydi=abs(sfydi); %Magnitude of spectrum of demodulator input signal
330

331 %
332
333 - fydo=fft(ydo); %Spectrum of demodulator output signal (Fund. Reg.)
334 - sfydo=fftshift(sfyo); %Spectrum of demodulator output signal (Princ. Reg.)
335 - asfydo=abs(sfyo); %Magnitude of spectrum of demodulator output signal
336
337 %
338
339 - fxr=fft(xr); %Spectrum of received signal (Fund. Reg.)
340 - sfxr=fftshift(sfxr); %Spectrum of received signal (Princ. Reg.)
341 - asfxr=abs(sfxr); %Magnitude of spectrum of received signal
342
343 %
344
345 - figure
346 - plot(fkq,asfs)
347 - xlabel('Frequency in Hertz')
348 - ylabel('Magnitude')
349 - title('Magnitude of Spectrum of Wanted Signal S(f)')
350 - grid
351
352 %
353
354 - figure
355 - plot(fkq,asfg)
356 - xlabel('Frequency in Hertz')
357 - ylabel('Magnitude')
358 - title('Magnitude of Spectrum of Interference Signal G(f)')
359 - grid
360

```

```

361 % %
362
363 - figure
364 - plot(fkq,asfxm)
365 - xlabel('Freq in Hertz')
366 - ylabel('Magnitude')
367 - title('Magnitude of Spectrum of Modulating Signal Xm(f)')
368 - grid
369 %
370 %
371
372 - figure
373 - plot(fkq,asfxc)
374 - xlabel('Freq in Hertz')
375 - ylabel('Magnitude')
376 - title('Magnitude of Spectrum of Modulated Signal Xc(f)')
377 - grid
378
379 %
380
381 - asfyci=transpose(asfyci);
382 - figure
383 - plot(fkqy,asfyci)
384 - xlabel('Frequency in Hertz')
385 - ylabel('Magnitude')
386 - title('Magnitude of Spectrum of Channel Input Signal Yci(f)')
387 - grid
388
389 %
390

```

```

391 -     asfyC=transpose(asfyC);
392 -     figure
393 -     plot(fkqyC,asfyC)
394 -     xlabel('Frequency in Hertz')
395 -     ylabel('Magnitude')
396 -     title('Magnitude of Spectrum of Channel Filter Output Signal Yc(f)')
397 -     grid
398 -
399 -     %
400 -
401 -     figure
402 -     plot(fkq,asfn)
403 -     xlabel('Frequency in Hertz')
404 -     ylabel('Magnitude')
405 -     title('Magnitude of Spectrum of AWGN Signal N(f)')
406 -     grid
407 -
408 -     %
409 -
410 -     asfyCo=transpose(asfyCo);
411 -     figure
412 -     plot(fkqyCo,asfyCo)
413 -     xlabel('Frequency in Hertz')
414 -     ylabel('Magnitude')
415 -     title('Magnitude of Spectrum of Channel Output Signal Yco(f)')
416 -     grid
417 -
418 -     %
419 -
420 -     asfyDi=transpose(asfyDi);

```

```

421 - figure
422 - plot(fkqydi,asfydi)
423 - xlabel('Frequency in Hertz')
424 - ylabel('Magnitude')
425 - title('Magnitude of Spectrum of Demodulator Input Signal Ydi(f)')
426 - grid
427
428 % %
429
430 - figure
431 - plot(fkqydo,asfydo)
432 - xlabel('Frequency in Hertz')
433 - ylabel('Magnitude')
434 - title('Magnitude of Spectrum of Demodulator Output Signal Ydo(f)')
435 - grid
436
437 % %
438
439 - figure
440 - plot(fkqxr,asfxr)
441 - xlabel('Frequency in Hertz')
442 - ylabel('Magnitude')
443 - title('Magnitude of Spectrum of Received Signal Xr(f)')
444 - grid
445
446 % %
447
448 - sound(st, Fs);
449 - pause(3);
450 - sound(xr, Fs);

```

**Code 6:** Matlab Script for Tasks 2-1

## Task 2-2:

```

1 %*****
2 %*****|*****Wanted Signal Input/Output*****|
3 %*****
4 - [st, Fss] = audioread('S086GP01_EDP_INP1s/dtmf01.wav'); % s(t) signal
5 - [gt, Fsg] = audioread('S086GP01_EDP_INP1g/uwtd01.wav'); % g(t) signal
6 - [nt,Fsn]=audioread('S086GP01_EDP_INP1n/nois01.wav'); % n(t) signal
7 %
8 %%****Processing Input Signals****%
9 %
10 save st01.txt st -ascii %Save signal as s.txt file
11 load -ascii st01.txt; %Load ascii file s.txt
12 ssiz=length(st); %Get the length of the "s" signal
13 %
14 save gt01.txt gt -ascii %Save signal as g.txt file
15 load -ascii gt01.txt; %Load ascii file g.txt
16 gsiz=length(gt); %Get the length of the "g" signal
17 %
18 save nt01.txt nt -ascii %Save signal as n.txt file
19 load -ascii nt01.txt; %Load ascii file n.txt
20 nsiz=length(nt); %Get the length of the "n" signal
21 %
22 %
23 %*****
24 %*****Parameter Settings*****%
25 %
26 - Fs=40000; %Sampling rate
27 - Ts=1/Fs;
28 - fc=8000; %Carrier frequency
29 - N=ssiz;
30 - V=N*Ts;

31 - fincq=1/V; %Frequency resolution of wanted signal
32 - fkq=-(Fs+1)/2:fincq:((Fs+1)/2)-fincq; %Frequency axis of wanted signal
33 -
34 - t=0:Ts:V-Ts;
35 - M=200; %Order of the filters
36 - Fv=2000;
37 -
38 %
39 %
40 % TRANSMITTER SYSTEM
41 %
42
43 - n = 3; %Number of bits
44 - maxst = max(st);
45 - i = 2*maxst/(2^n -1); %Interval length
46 - mxst = maxst + i; %One interval above max value (to have intervals of [-max,max])
47 - partition = [-maxst:i:mxst]; %Quantization intervals
48 - codebook = [-maxst:i:mxst]; %Value quantizer assigns
49 - [index, quants] = quantiz(st, partition,codebook);
50 - st = transpose(quants);
51 - audiowrite("S086GP01_EDP_INP2s/xmt01.wav",st,Fs);

52
53 - maxgt = max(gt); %Interval length
54 - i = 2*maxgt/(2^n -1); %One interval above max value (to have intervals of [-max,max])
55 - mxgt = maxgt + i; %Quantization intervals
56 - partition = [-maxgt:i:mxgt];
57 - codebook = [-maxgt:i:mxgt]; %Value quantizer assigns
58 - [index, quants] = quantiz(gt, partition,codebook);
59 - gt = transpose(quants);
60 - audiowrite("S086GP01_EDP_INP2g/xmt01.wav",gt,Fs);

```

```

61
62 - c=cos(2*pi*fct);
63 - xm=st+gt;
64 - audiowrite("S086GP01_EDP_INP2aq/xmt01.wav",xm,Fs);
65 - cl=transpose(c);
66 - xc=xm.*cl;
67 - %Modulated wanted signal plus interference
68
69 %% Part #1 task #1
70 %% BAND_PASS FILTER DESIGN (hB)
71 %%
72 - fm=Fv;
73 - fUpper = fc + fm;
74 - fLower = fc - fm;
75 - Wnl = 2*(fLower)/Fs;
76 - Wnu = 2*(fUpper)/Fs;
77 - Fb = [Wnl Wnu];
78 - hB = fir1(M-1,Fb,'bandpass');
79 - yci=conv(xc,hB,'same');
80 - %band limits
81 - %filter design
82 - %Removal of unwanted interference signals
83
84 - ycisiz=length(yci);
85 - Nyci=ycisiz;
86 - Vyci=Nyci*Ts;
87 - fincqyci=1/Vyci;
88 - %Frequency resolution of channel input signal
89 - fkqyci=-( (Fs+1)/2 ) : fincqyci : +((Fs+1)/2)-fincqyci; %Frequency axis of channel input signal
90
91 %% Part #1 task #2
92 %% LOW-PASS FILTER DESIGN (Tc)
93
94 - %% BAND_PASS FILTER DESIGN (hC)
95 - B = 12000;
96 - M=100;
97 - Wn=2*B/Fs;
98 - th0=Ts:M*Ts-Ts;
99 - hc=fir1(M-1,Wn);
100 - yc=conv(yci,hc,'same');
101 - %Normilized frequency
102 - %Time axis to plot impulse response signal
103 - %Impulse response signal
104 - ycsiz=length(yc);
105 - Nyc=ycsiz;
106 - Vyc=Nyc*Ts;
107 - fincqyc=1/Vyc;
108 - %Frequency resolution of channel filter output signal
109 - fkqyc=-( (Fs+1)/2 ) : fincqyc : +((Fs+1)/2)-fincqyc; %Frequency axis of channel filter output signal
110
111 %% Part #1 task #3
112 %% ADDITION OF NOISE SIGNAL (Tn)
113 - %% ADDITION OF NOISE SIGNAL (Tn)
114 - yco = yc+nt;
115 - %Frequency resolution of channel output signal
116 - fkqyco=-( (Fs+1)/2 ) : fincqyco : +((Fs+1)/2)-fincqyco; %Frequency axis of channel output signal
117 - xmSNR = snr(st,gt);
118 - ycPdB = pow2db(bandpower(yc));
119 - nPdB = pow2db(bandpower(nt));
120 - SNR = snr(yc,nt);

```

```

121 - nsig_avg = mean(nt)                                %Mean of noise
122 - nsig_var = var(nt)                               %Variance of noise
123
124 %% Part #1 task #4
125 %%
126 %% BANDPASS FILTER (Tr)
127 %%
128 - fR = [Wnl Wnu];                                 %band limits
129 - hR = fir1(15,fR,'bandpass');                   %filter design
130 - ydi = conv(yco,hR,'same');                     %Removal of unwanted interference signals
131
132 - ydisiz = length(ydi);
133 - Nydi = ydisiz;
134 - Vydi = Nydi*Ts;
135 - fincqydi = 1/Vydi;                             %Frequency resolution of demodulator input signal
136 - fkqydi = -(Fs+1)/2:fincqydi:((Fs+1)/2)-fincqydi; %Frequency axis of demodulator input signal
137
138 - ydiPdB = pow2db(bandpower(ydi));
139 - SNR2 = snr(ydi-nt,nt)
140
141 %%***** Demodulator Td*****
142 - ydo = ydi.* (2*c1);
143
144 - ydosiz=length(ydo);
145 - Nydo=ydosiz;
146 - Vydo=Nydo*Ts;
147 - fincqydo=1/Vydo;                            %Frequency resolution of demodulator output signal
148 - fkqydo=-(Fs+1)/2:fincqydo:((Fs+1)/2)-fincqydo; %Frequency axis of demodulator output signal
149
150 - ...
151 %%
152 %% LOW_PASS FILTER DESIGN (Tr)
153 %%
154 - Wn=2*fm/Fs;          %Normalized cut-off frequency
155 - M=50;                %Order of the impulse response signal hL(t)
156 - thL=0:Ts:M*Ts-Ts;   %Time axis to plot impulse response signal
157 - hL=fir1(M-1,Wn);    %Impulse response signal
158 - xr=conv(ydo,hL);    %Convolution operation for filtering
159
160 - audiowrite("S086GP01_EDP_OUP202/dtmf01.wav",xr,Fs);
161
162 - Dr=((M-1)/2)*Ts
163 - xrsiz=length(xr);
164 - Nxr=xrsiz;
165 - Vxr=Nxr*Ts;
166 - fincqxr=1/Vxr;           %Frequency resolution of received signal
167 - fkqxr=-(Fs+1)/2:fincqxr:((Fs+1)/2)-fincqxr; %Frequency axis of received signal
168
169 %%
170 %% *****PLOTS*****
171 %%
172 - tyci=0:Ts:(length(yci)-1)*Ts;
173 - tyc=0:Ts:(length(yc)-1)*Ts;
174 - txr=0:Ts:(length(xr)-1)*Ts;
175
176
177 - % % Time Domain
178
179 - plot(t,st);
180 - axis([0 (length(st)*Ts) min(st) max(st)]);

```

```
181 - xlabel('Time in Seconds')
182 - ylabel('Amplitude')
183 - title('Wanted signal s(t)')
184 - grid
185
186 % %
187
188 - figure
189 - plot(t,gt);
190 - axis([0 (length(gt)*Ts) min(gt) max(gt)]);
191 - xlabel('Time in Seconds')
192 - ylabel('Amplitude')
193 - title('Interference Signal g(t)')
194 - grid
195
196 % %
197
198 - figure
199 - plot(t,xm)
200 - xlabel('Time in Seconds')
201 - ylabel('Amplitude')
202 - title('Modulating Signal (xm(t) = s(t) + g(t))')
203 - grid
204
205 % %
206
207 - figure
208 - plot(t,xc)
209 - xlabel('Time in Seconds')
210 - ylabel('Amplitude')
```

```

211 - title('Modulated Signal xc(t)')
212 - grid
213
214 % %
215
216 - figure
217 - plot(tyci,yci)
218 - axis([0, (length(yci)*Ts), min(yci), max(yci)]);
219 - xlabel('Time in Seconds')
220 - ylabel('Amplitude')
221 - title('Channel Input Signal yci(t)')
222 - grid
223
224 % %
225
226 - figure
227 - plot(tyci,yc)
228 - axis([0 (length(yc)*Ts) min(yc) max(yc)]);
229 - xlabel('Time in Seconds')
230 - ylabel('Amplitude')
231 - title('Channel Filter Output Signal yc(t)')
232 - grid
233
234 % %
235
236 - figure
237 - plot(tyc,nt)
238 - axis([0 (length(nt)*Ts) min(nt) max(nt)]);
239 - xlabel('Time in Seconds')
240 - ylabel('Amplitude')

```

```

241 - title('Channel Noise Signal n(t)')
242 - grid
243
244 % %
245
246 - figure
247 - plot(tyc,yco)
248 - axis([0 (length(yco)*Ts) min(yco) max(yco)]);
249 - xlabel('Time in Seconds')
250 - ylabel('Amplitude')
251 - title('Channel Output Signal yco(t)')
252 - grid
253
254 % %
255
256 - figure
257 - plot(tyc,ydi)
258 - axis([0 (length(ydi)*Ts) min(ydi) max(ydi)]);
259 - xlabel('Time in Seconds')
260 - ylabel('Amplitude')
261 - title('Demodulator Input Signal ydi(t)')
262 - grid
263
264 % %
265
266 - figure
267 - plot(tyc,ydo)
268 %axis([0, (length(ydo)*Ts), min(ydo), max(ydo)]); ****
269 - xlabel('Time in Seconds')
270 - ylabel('Amplitude')

```

```

271 - title('Demodulator Output Signal ydo(t)')
272 - grid
273
274 % %
275
276 - figure
277 - plot(txr,xr)
278 %axis([0 (length(xr)*Ts) min(xr) max(xr)]);
279 - xlabel('Time in Seconds')
280 - ylabel('Amplitude')
281 - title('Received Signal xr(t)')
282 - grid
283
284 % % Spectrums
285
286 - fw=fft(st); %Spectrum of wanted signal (Fund. Reg.)
287 - sfs=fftshift(fw); %Spectrum of wanted signal (Princ. Reg.)
288 - asfs=abs(sfs); %Magnitude of spectrum of wanted signal
289
290 %
291
292 - fg=fft(gt); %Spectrum of interference signal (Fund. Reg.)
293 - sfg=fftshift(fg); %Spectrum of interference signal (Princ. Reg.)
294 - asfg=abs(sfg); %Magnitude of spectrum of wanted signal
295
296 %
297
298 - fxm=fft(xm); %Spectrum of input signal (Fund. Reg.)
299 - sfxm=fftshift(fxm); %Spectrum of input signal (Princ. Reg.)
300 - asfxm=abs(sfxm); %Magnitude of spectrum of input signal

```

---

```

301 %
302
303
304 - fxc=fft(xc); %Spectrum of modulated signal (Fund. Reg.)
305 - sfxc=fftshift(fxc); %Spectrum of modulated signal (Princ. Reg.)
306 - asfxc=abs(sfxc); %Magnitude of spectrum of modulated signal
307
308 %
309
310 - fyci=fft(yci); %Spectrum of channel input signal (Fund. Reg.)
311 - sfyci=fftshift(fyci); %Spectrum of channel input signal (Princ. Reg.)
312 - asfyci=abs(sfyci); %Magnitude of spectrum of channel input signal
313
314 %
315
316 - fyc=fft(yc); %Spectrum of channel filter output signal (Fund. Reg.)
317 - sfyc=fftshift(fyc); %Spectrum of channel filter output signal (Princ. Reg.)
318 - asfyc=abs(sfyc); %Magnitude of spectrum of channel filter output signal
319
320 %
321
322 - fn=fft(nt); %Spectrum of AWGN signal (Fund. Reg.)
323 - sfn=fftshift(fn); %Spectrum of AWGN signal (Princ. Reg.)
324 - asfn=abs(sfn); %Magnitude of spectrum of wanted signal
325
326 %
327
328 - fyco=fft(yco); %Spectrum of channel output signal (Fund. Reg.)
329 - sfyco=fftshift(fyco); %Spectrum of channel output signal (Princ. Reg.)
330 - asfyco=abs(sfyco); %Magnitude of spectrum of channel output signal
331
332 %
333
334 - fydi=fft(ydi); %Spectrum of demodulator input signal (Fund. Reg.)
335 - sfydi=fftshift(fydi); %Spectrum of demodulator input signal (Princ. Reg.)
336 - asfydi=abs(sfydi); %Magnitude of spectrum of demodulator input signal
337
338 %
339
340 - fydo=fft(ydo); %Spectrum of demodulator output signal (Fund. Reg.)
341 - sfydo=fftshift(fydo); %Spectrum of demodulator output signal (Princ. Reg.)
342 - asfydo=abs(sfydo); %Magnitude of spectrum of demodulator output signal
343
344 %
345
346 - fxr=fft(xr); %Spectrum of received signal (Fund. Reg.)
347 - sfxr=fftshift(fxr); %Spectrum of received signal (Princ. Reg.)
348 - asfxr=abs(sfxr); %Magnitude of spectrum of received signal
349
350 %
351
352 - figure
353 - plot(fkq,asfs)
354 - xlabel('Frequency in Hertz')
355 - ylabel('Magnitude')
356 - title('Magnitude of Spectrum of Wanted Signal S(f)')
357 - grid
358
359 % %
360

```

```

361 - figure
362 - plot(fkq,asfg)
363 - xlabel('Frequency in Hertz')
364 - ylabel('Magnitude')
365 - title('Magnitude of Spectrum of Interference Signal G(f)')
366 - grid
367
368 % %
369
370 - figure
371 - plot(fkq,asfxm)
372 - xlabel('Freq in Hertz')
373 - ylabel('Magnitude')
374 - title('Magnitude of Spectrum of Modulating Signal Xm(f)')
375 - grid
376
377 % %
378
379 - figure
380 - plot(fkq,asfxc)
381 - xlabel('Freq in Hertz')
382 - ylabel('Magnitude')
383 - title('Magnitude of Spectrum of Modulated Signal Xc(f)')
384 - grid
385
386 % %
387
388 - asfyci=transpose(asfyci);
389 - figure
390 - plot(fkqy,asfy)

```

```

391 - xlabel('Frequency in Hertzs')
392 - ylabel('Magnitude')
393 - title('Magnitude of Spectrum of Channel Input Signal Yci(f)')
394 - grid
395
396 % %
397
398 - asfycc=transpose(asfycc);
399 - figure
400 - plot(fkqyc,asfycc)
401 - xlabel('Frequency in Hertzs')
402 - ylabel('Magnitude')
403 - title('Magnitude of Spectrum of Channel Filter Output Signal Yc(f)')
404 - grid
405
406 % %
407
408 - figure
409 - plot(fkq,asfn)
410 - xlabel('Frequency in Hertzs')
411 - ylabel('Magnitude')
412 - title('Magnitude of Spectrum of AWGN Signal N(f)')
413 - grid
414
415 % %
416
417 - asfyco=transpose(asfyco);
418 - figure
419 - plot(fkqyco,asfyco)
420 - xlabel('Frequency in Hertzs')

```

```

421 - ylabel('Magnitude')
422 - title('Magnitude of Spectrum of Channel Output Signal Yco(f)')
423 - grid
424
425 % %
426
427 - asfydi=transpose(asfydi);
428 - figure
429 - plot(fkqydi,asfydi)
430 - xlabel('Frequency in Hertz')
431 - ylabel('Magnitude')
432 - title('Magnitude of Spectrum of Demodulator Input Signal Ydi(f)')
433 - grid
434
435 % %
436
437 - figure
438 - plot(fkqydo,asfydo)
439 - xlabel('Frequency in Hertz')
440 - ylabel('Magnitude')
441 - title('Magnitude of Spectrum of Demodulator Output Signal Ydo(f)')
442 - grid
443
444 % %
445
446 - figure
447 - plot(fkqxrxr,asfxr)
448 - xlabel('Frequency in Hertz')
449 - ylabel('Magnitude')
450 - title('Magnitude of Spectrum of Received Signal Xr(f)')

451 - grid
452
453 % %
454
455 - sound(st, Fs);
456 - pause(3);
457 - sound(xr, Fs);

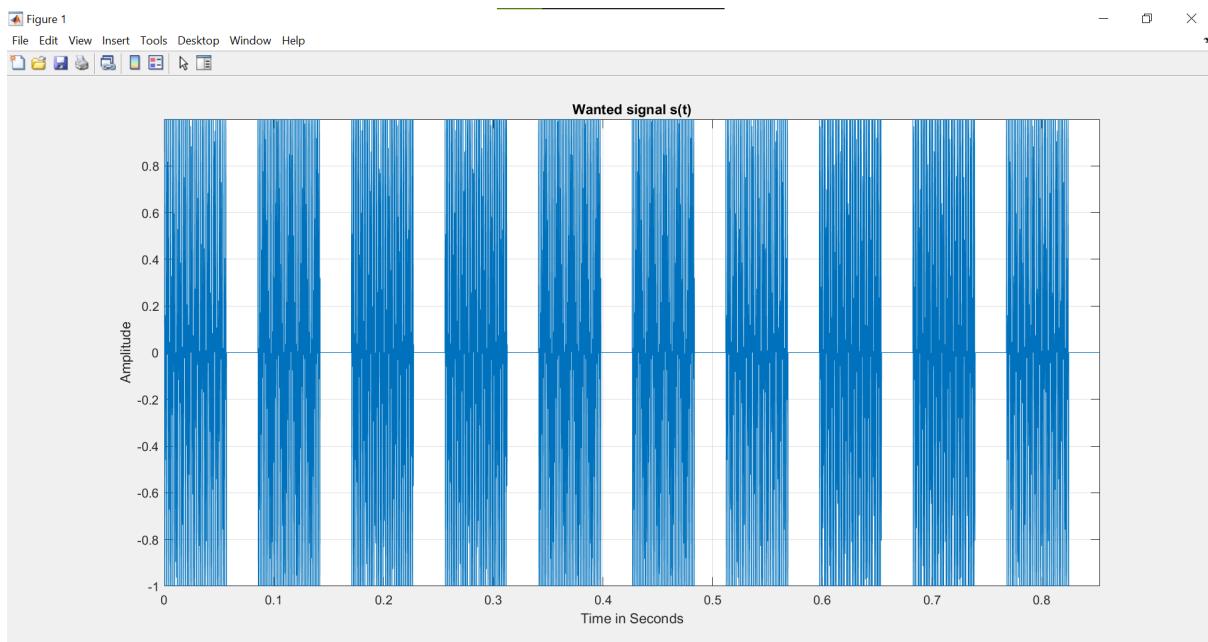
```

---

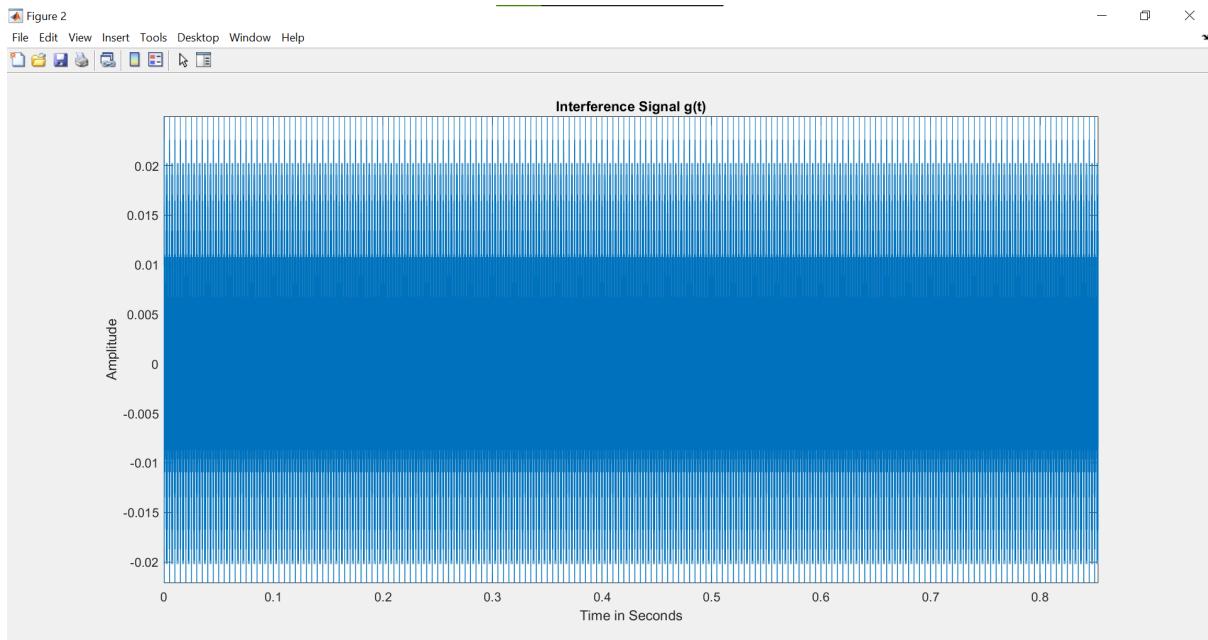
**Code 7:** Matlab Script for Tasks 2-2

## G. Graphs

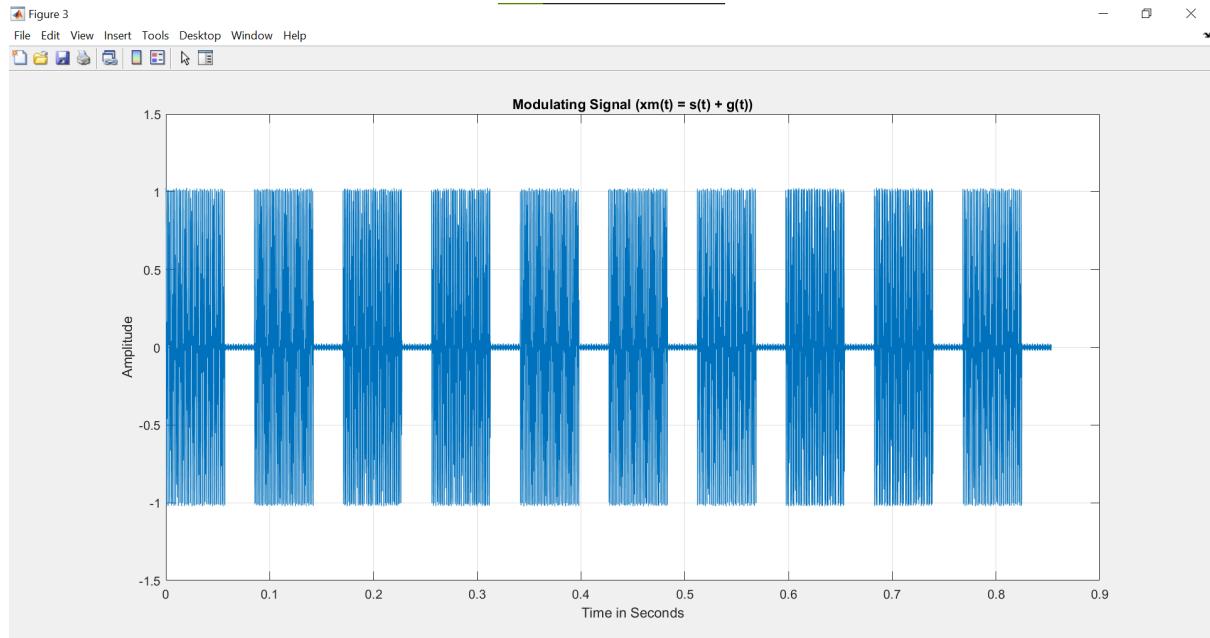
### Tasks 1-5:



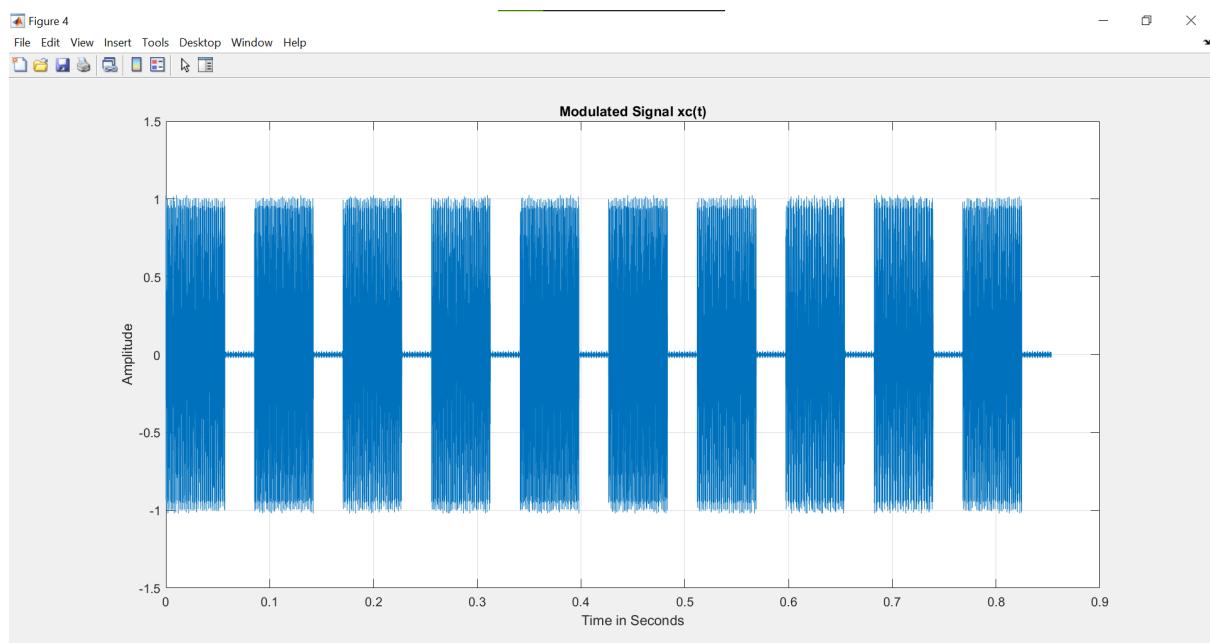
**Figure 1: Wanted Signal  $s(t)$  in the Time Domain.**



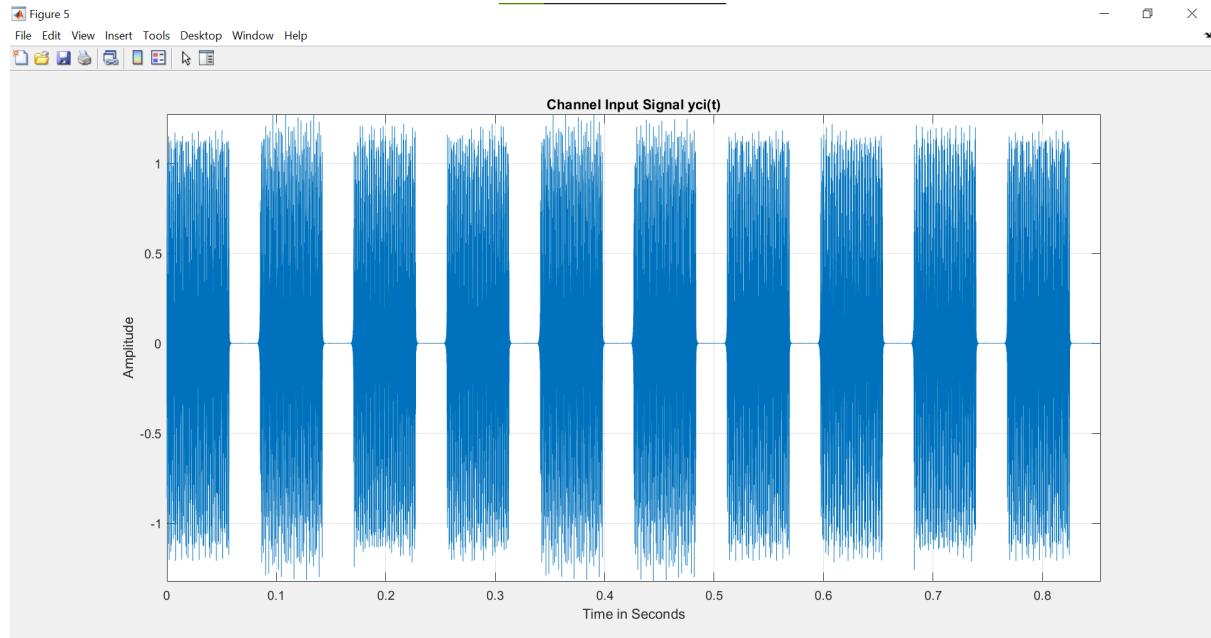
**Figure 2: Interference Signal  $g(t)$  in the Time Domain.**



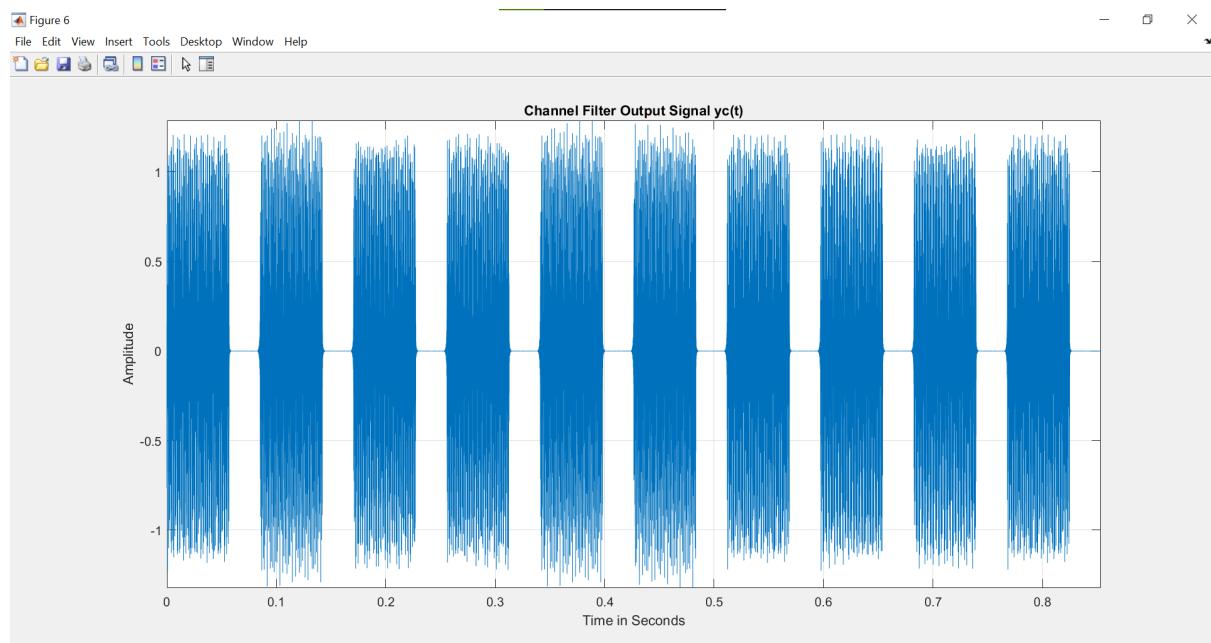
**Figure 3:** Modulating Signal ( $xm(t) = s(t) * g(t)$ )



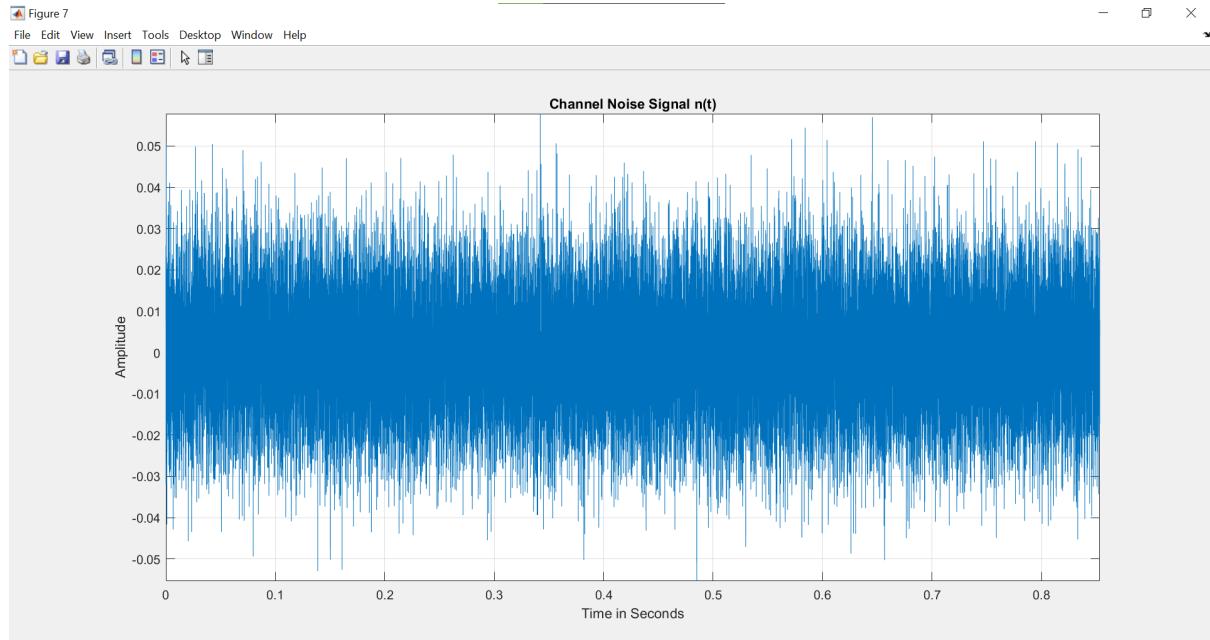
**Figure 4:** Modulated Signal  $xc(t)$  in the Time Domain



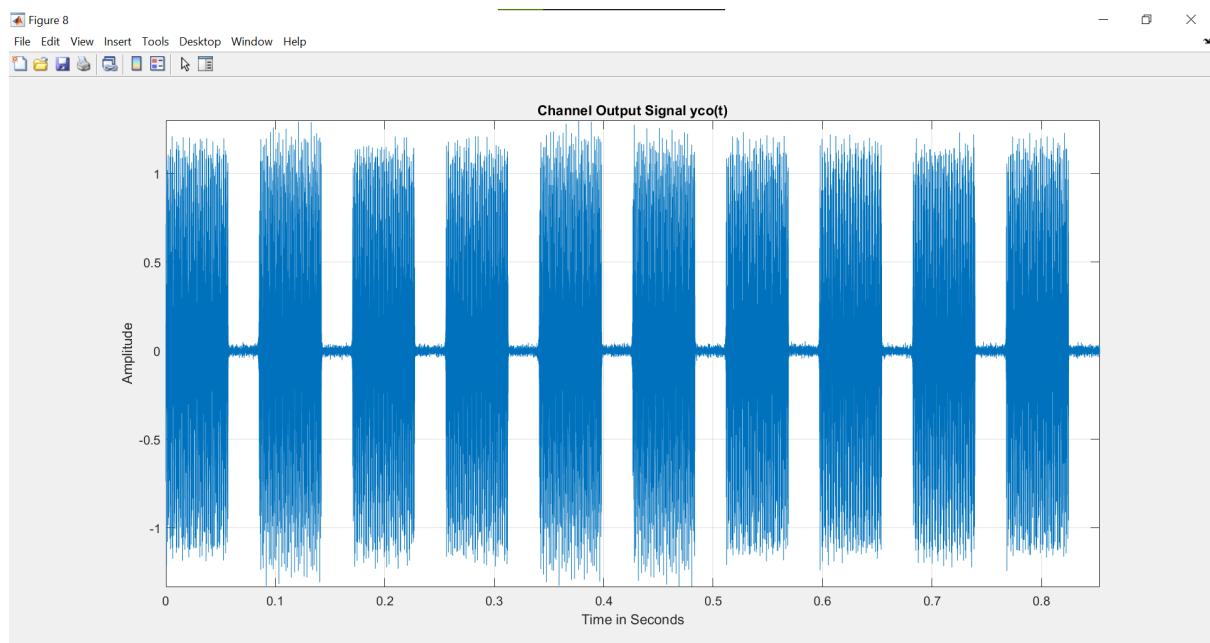
**Figure 5:** Channel Input Signal  $y_{ci}(t)$



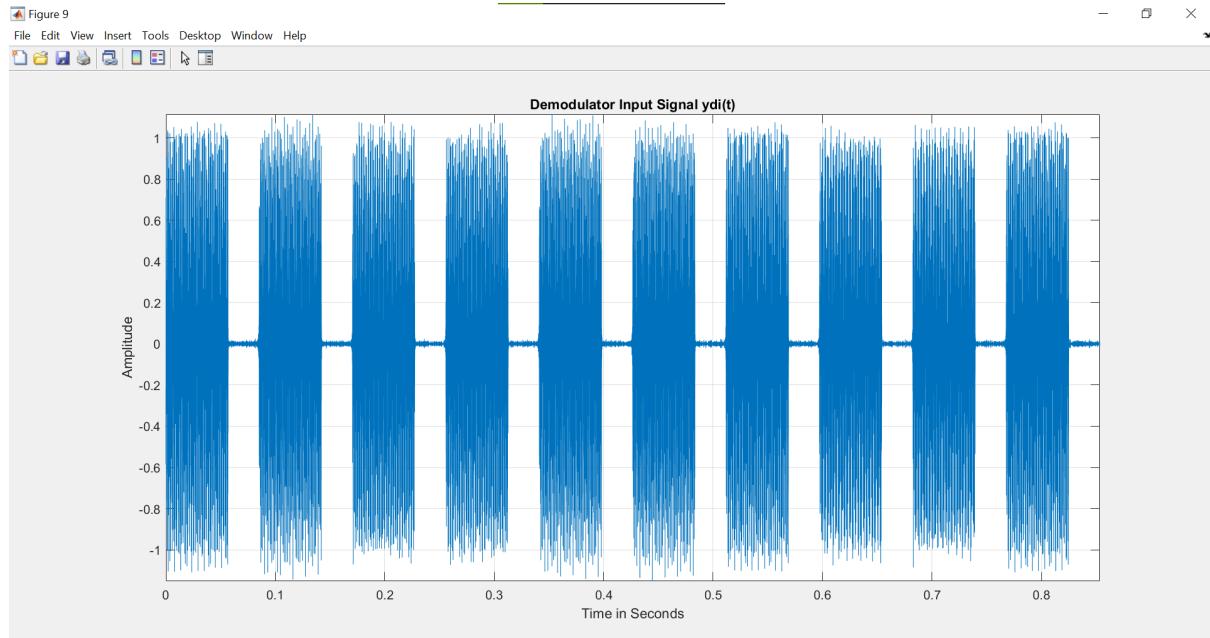
**Figure 6:** Channel Filter Output Signal  $y_c(t)$



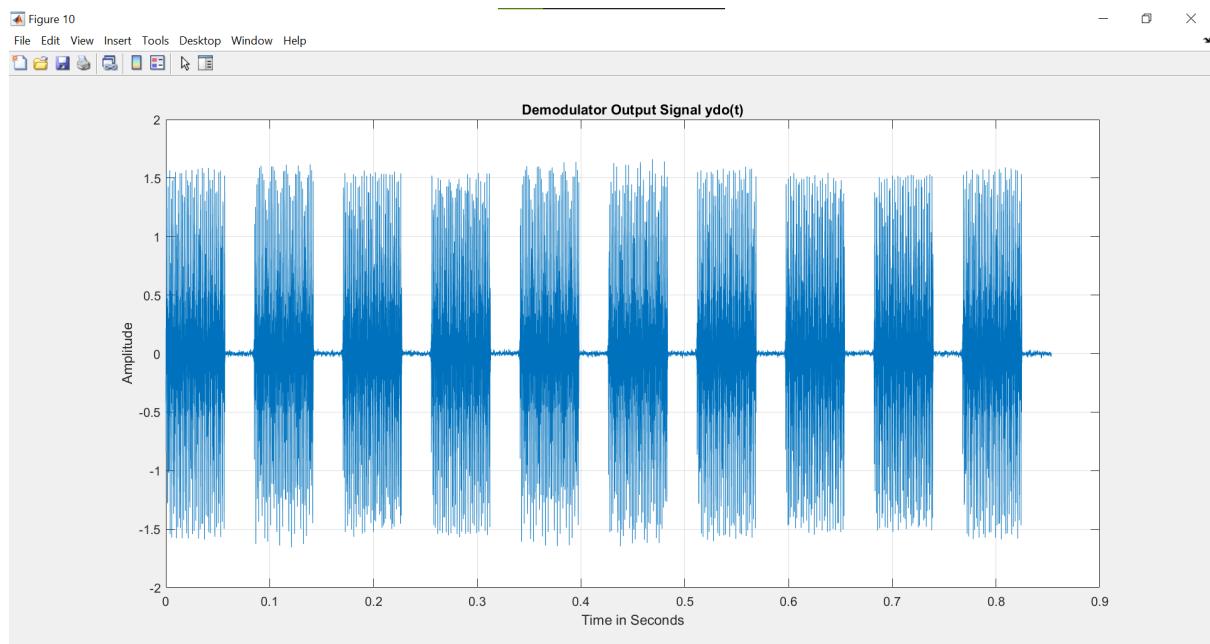
**Figure 7: Channel Noise Signal  $n(t)$**



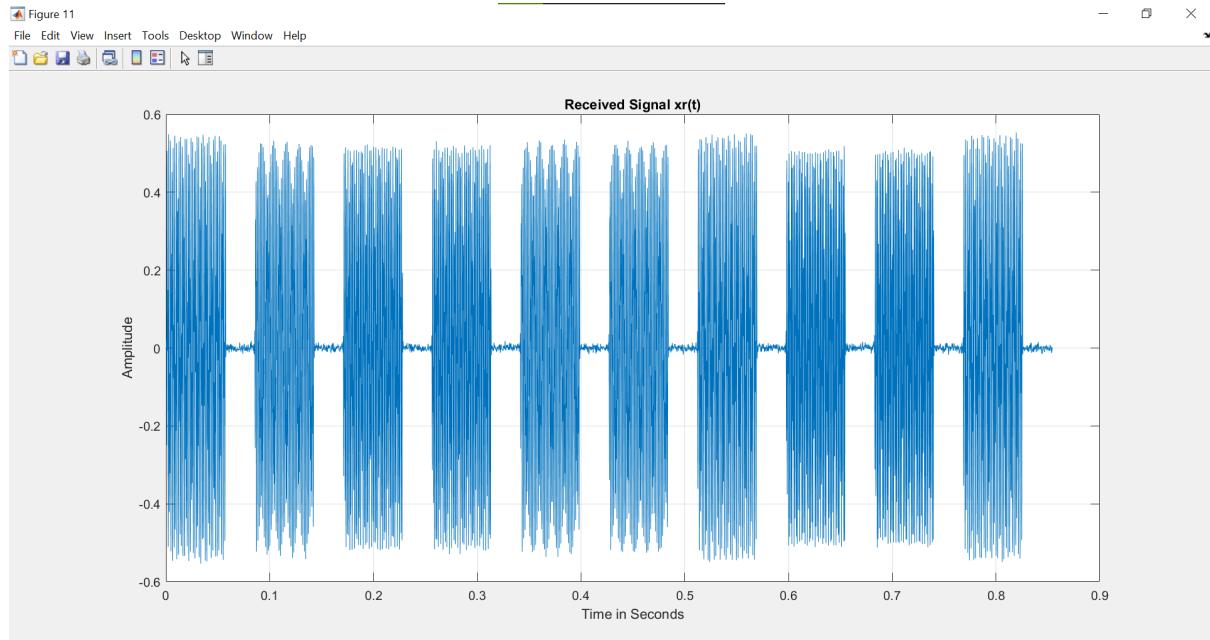
**Figure 8: Channel Output Signal  $y_{co}(t)$**



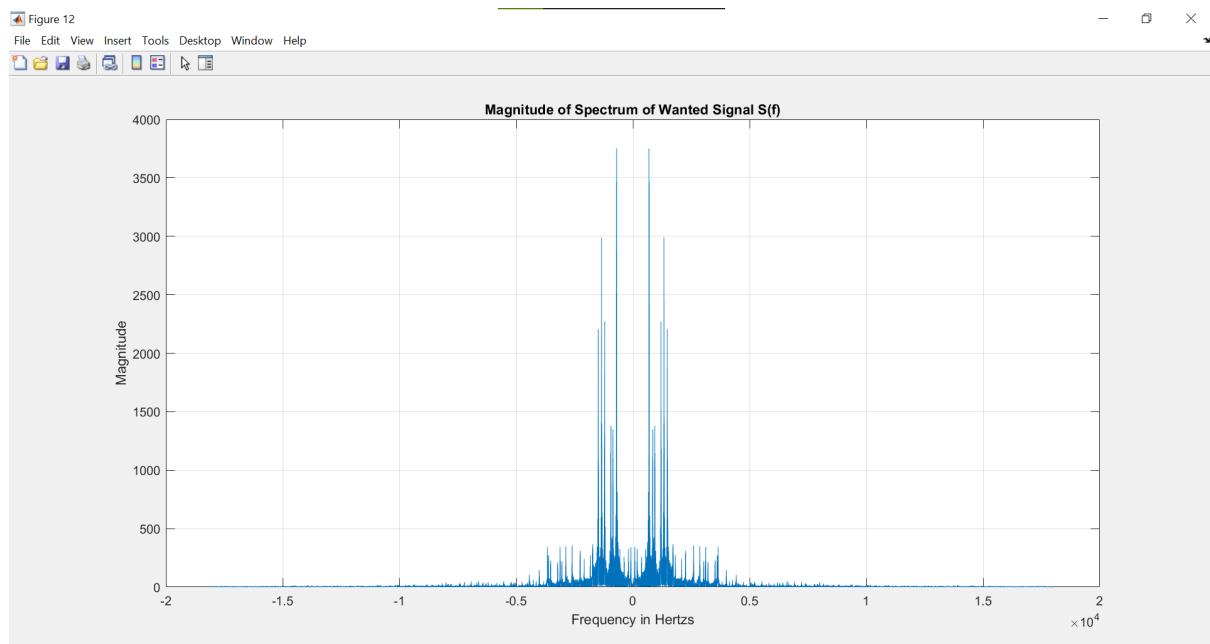
**Figure 9:** Demodulator Input Signal  $y_{di}(t)$



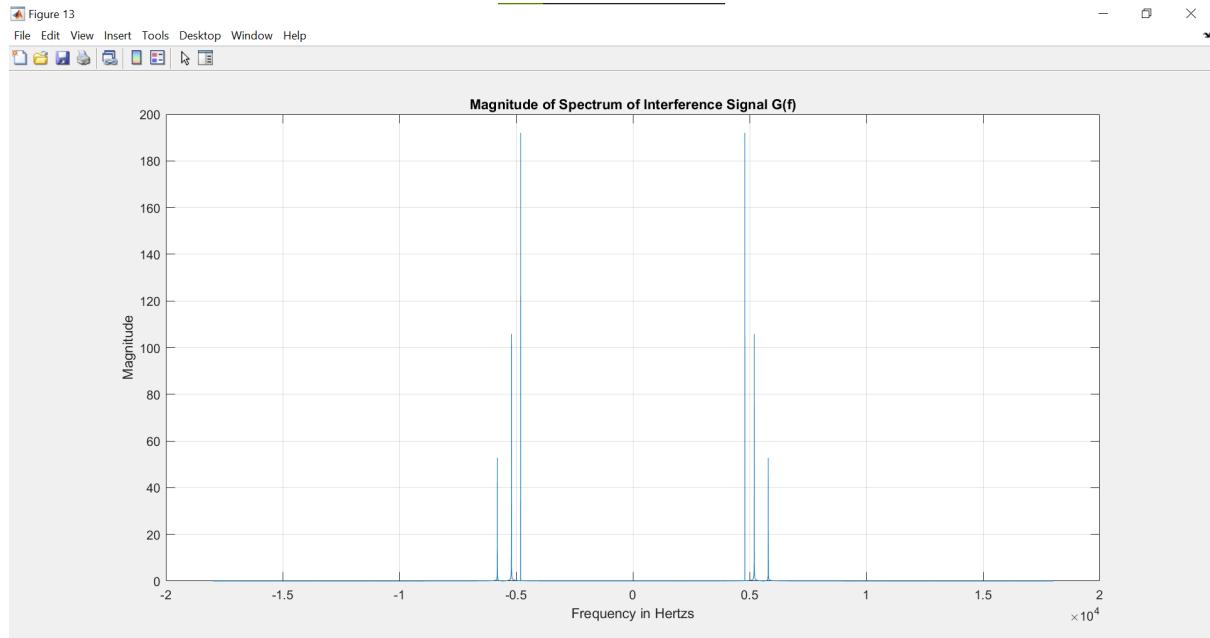
**Figure 10:** Demodulator Output Signal  $y_{do}(t)$



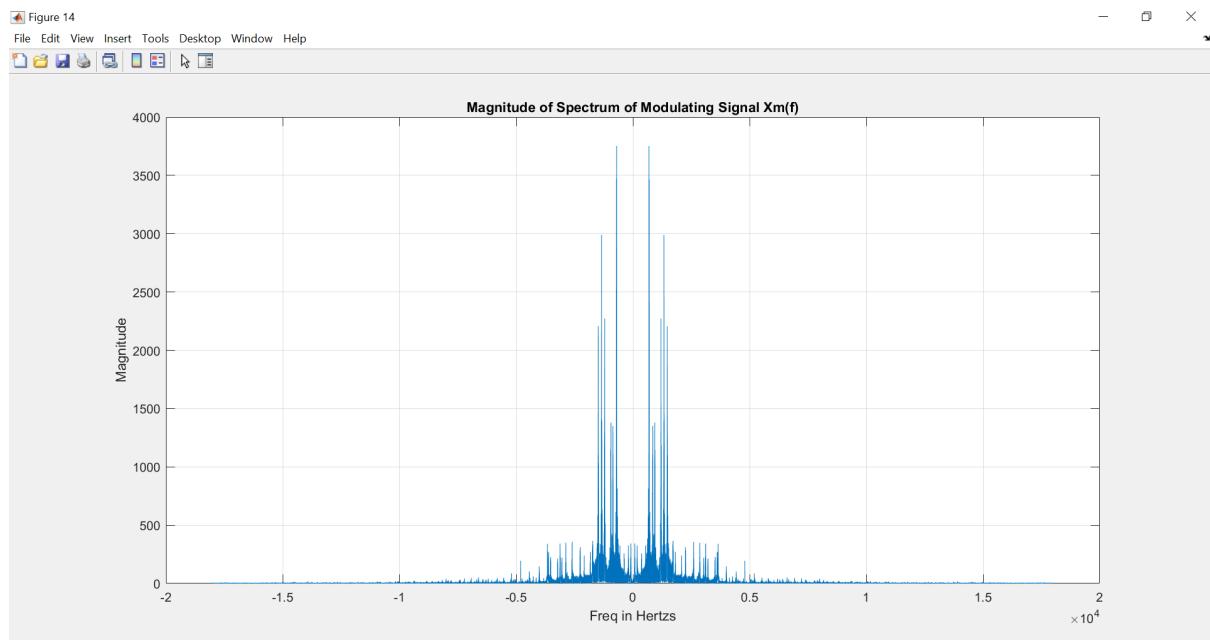
**Figure 11:** Received Signal  $x_r(t)$



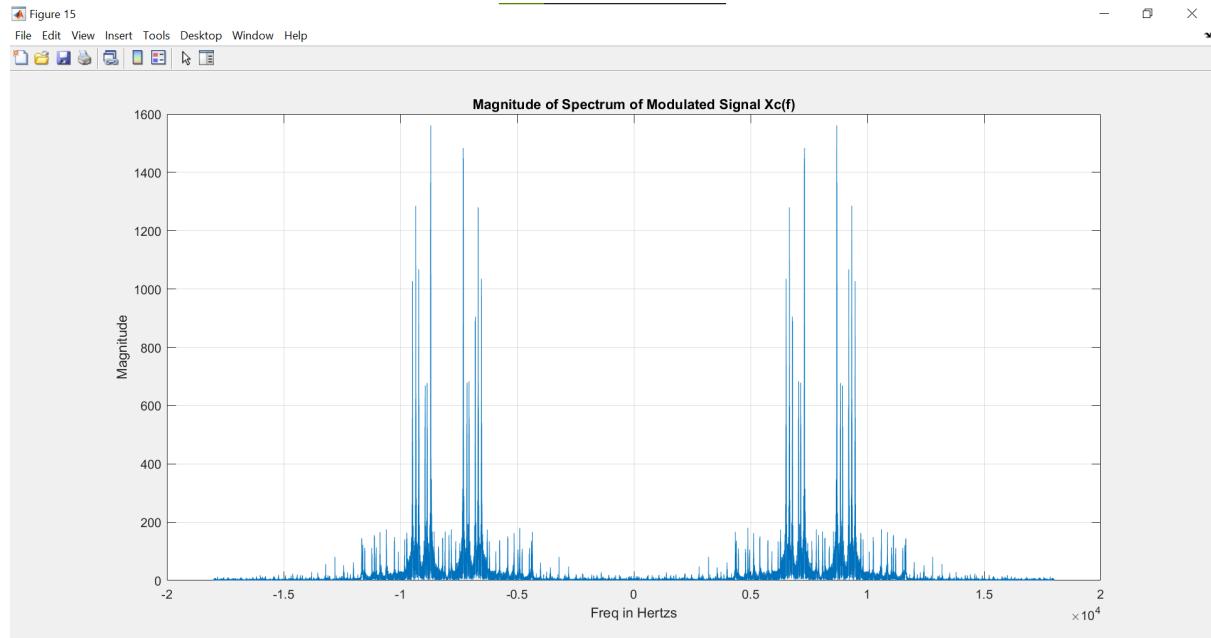
**Figure 12:** Magnitude of Spectrum of Wanted Signal  $S(f)$



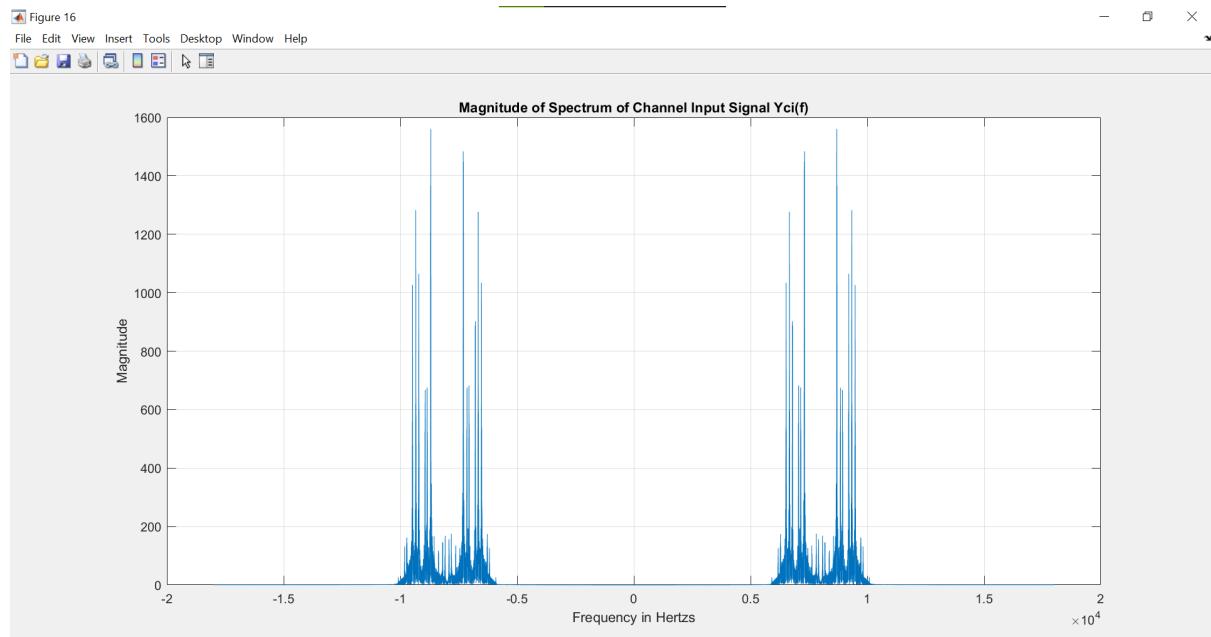
**Figure 13: Magnitude of Spectrum of modulated signal**



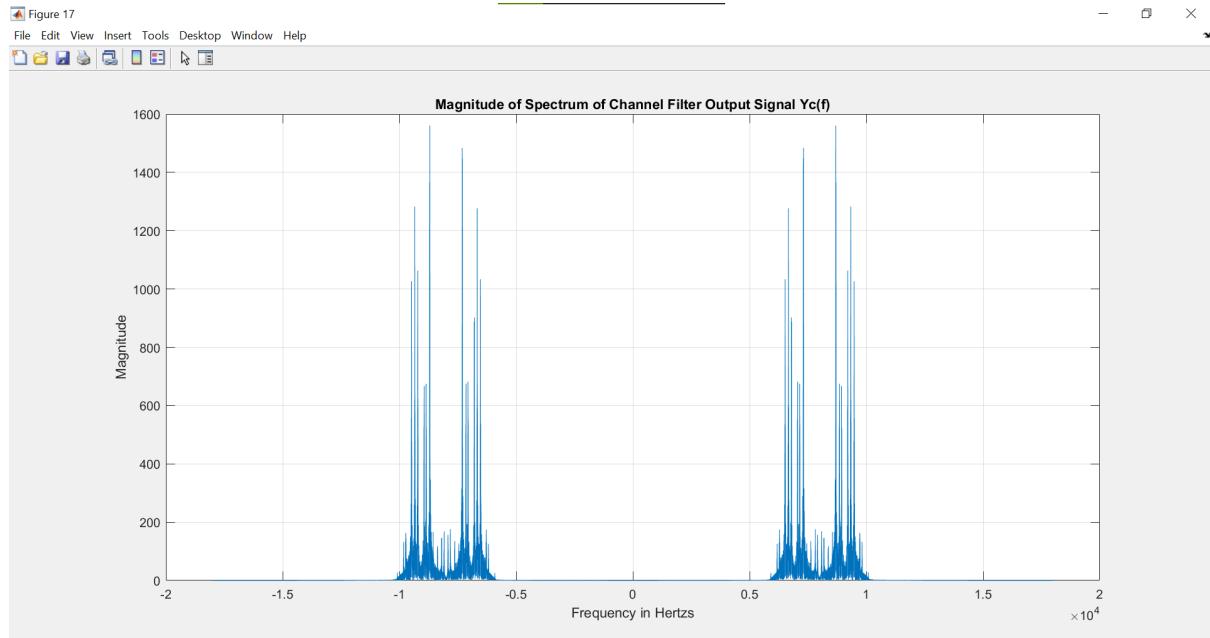
**Figure 14: Magnitude of Spectrum of Modulating Signal  $X_m(f)$**



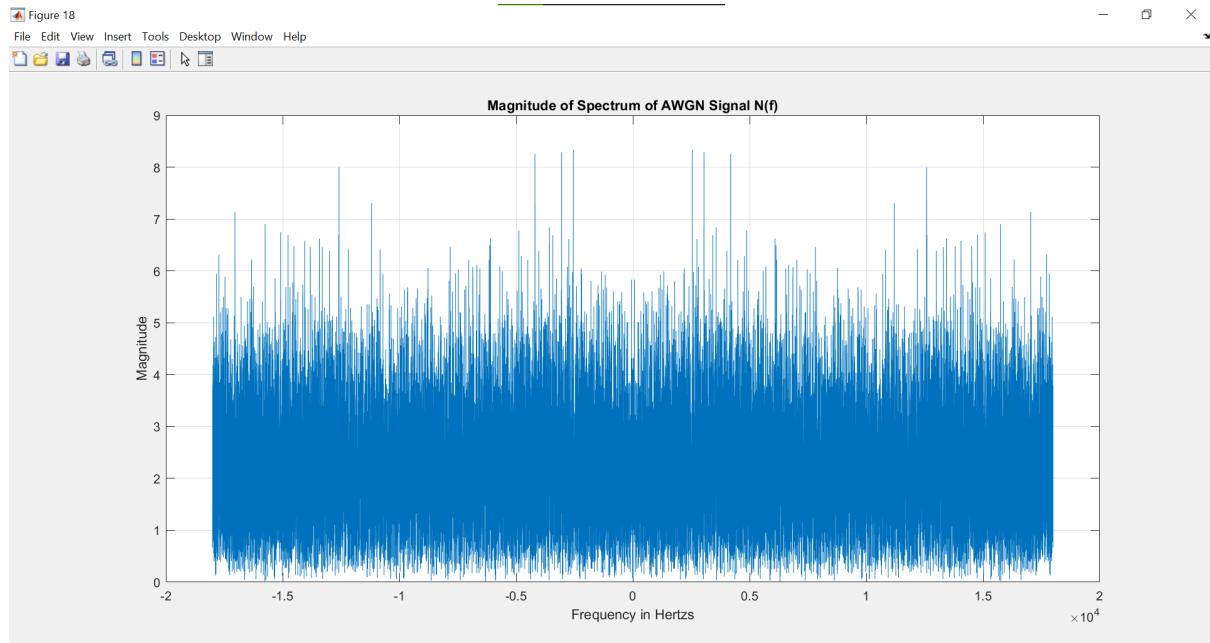
**Figure 15: Magnitude of Spectrum Modulating Signal  $X_c(f)$**



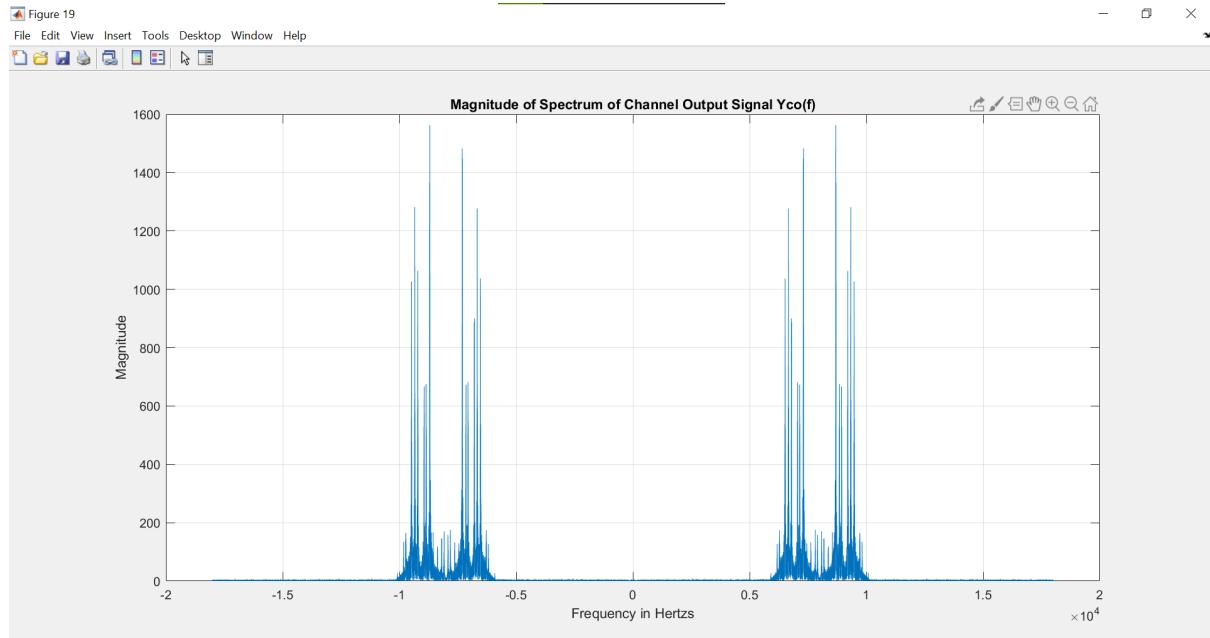
**Figure 16: Magnitude of Spectrum Channel Input Signal  $Y_{ci}(f)$**



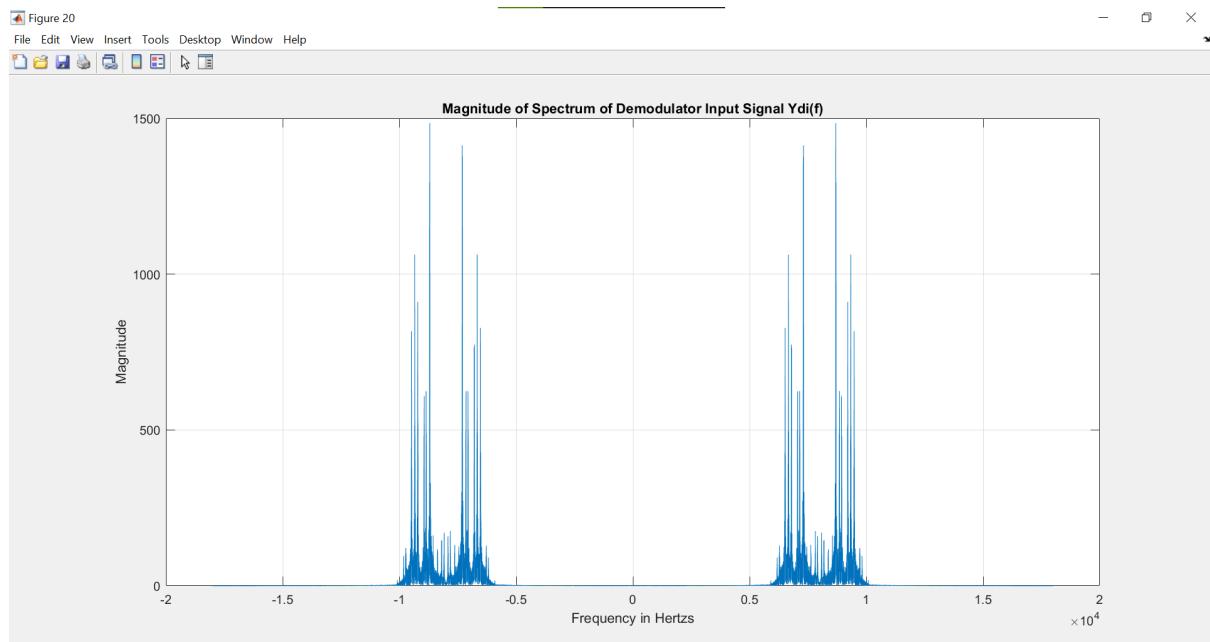
**Figure 17: Magnitude of Spectrum of Channel Filter Output Signal  $Y_c(f)$**



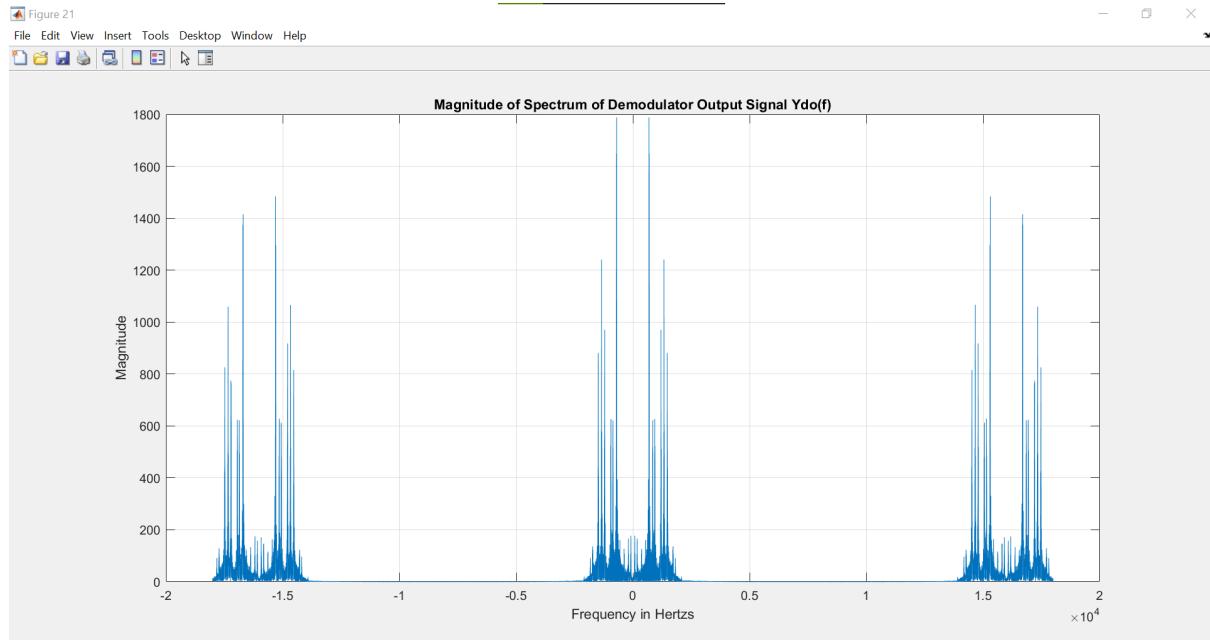
**Figure 18: Magnitude of Spectrum of AWGN Signal  $N(f)$**



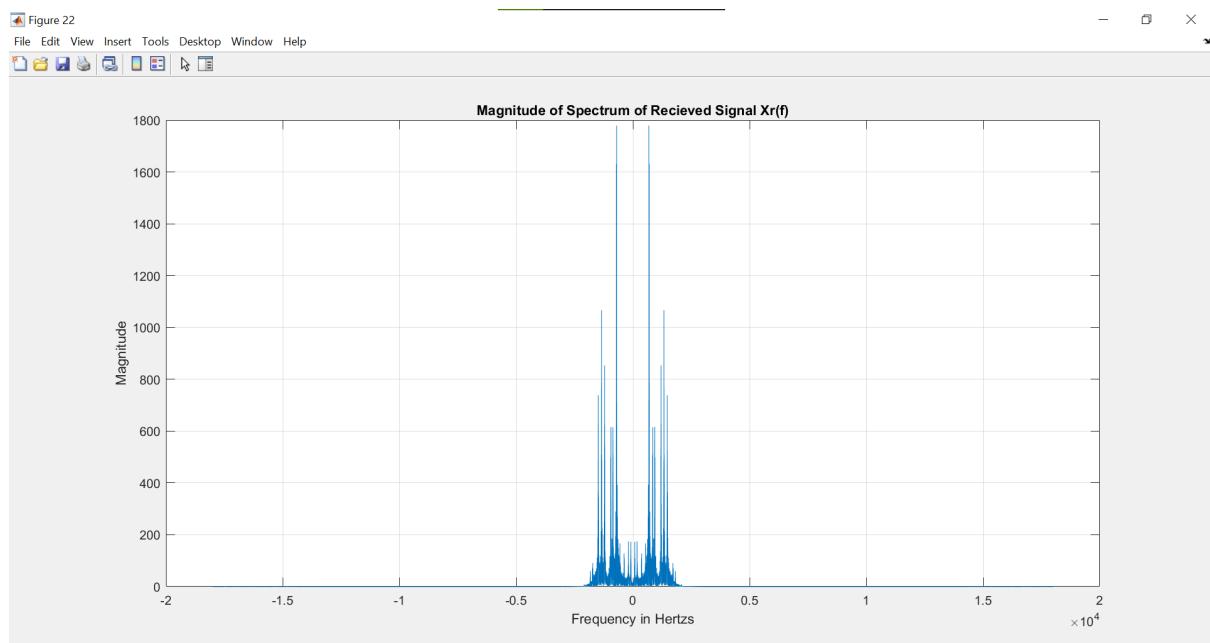
**Figure 19:** Magnitude of Spectrum of Channel Output Signal  $Y_{co}(f)$



**Figure 20:** Magnitude of Spectrum of Demodulator Input Signal  $Y_{di}(f)$

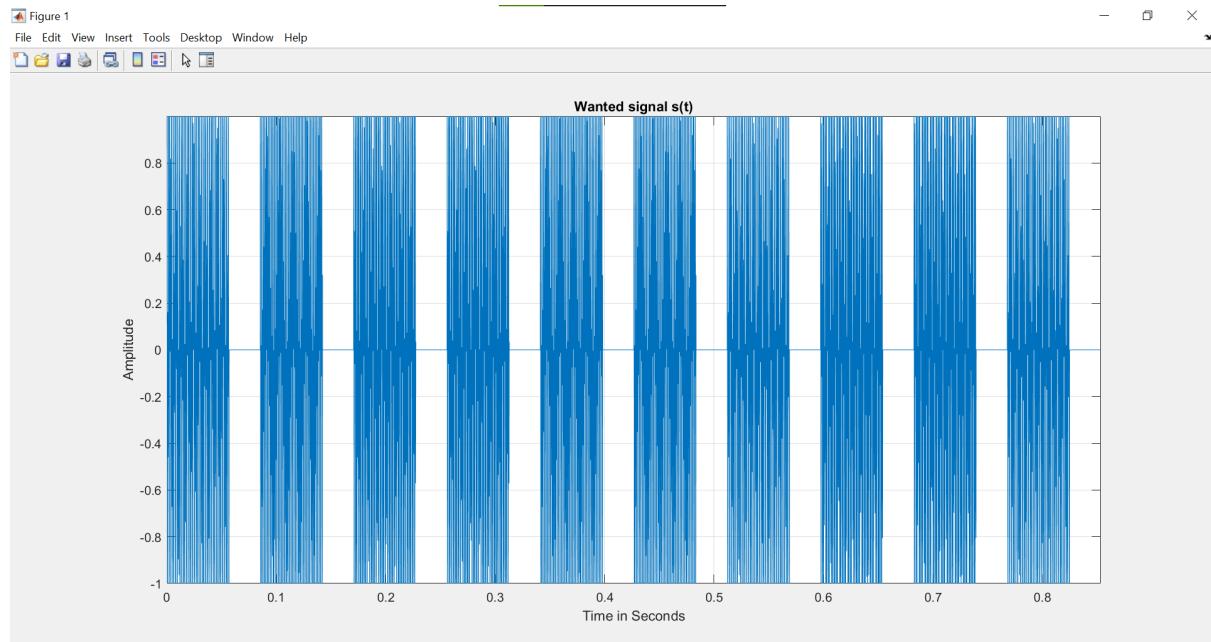


**Figure 21:** Magnitude of Spectrum of Demodulator Output Signal  $Y_{do}(f)$

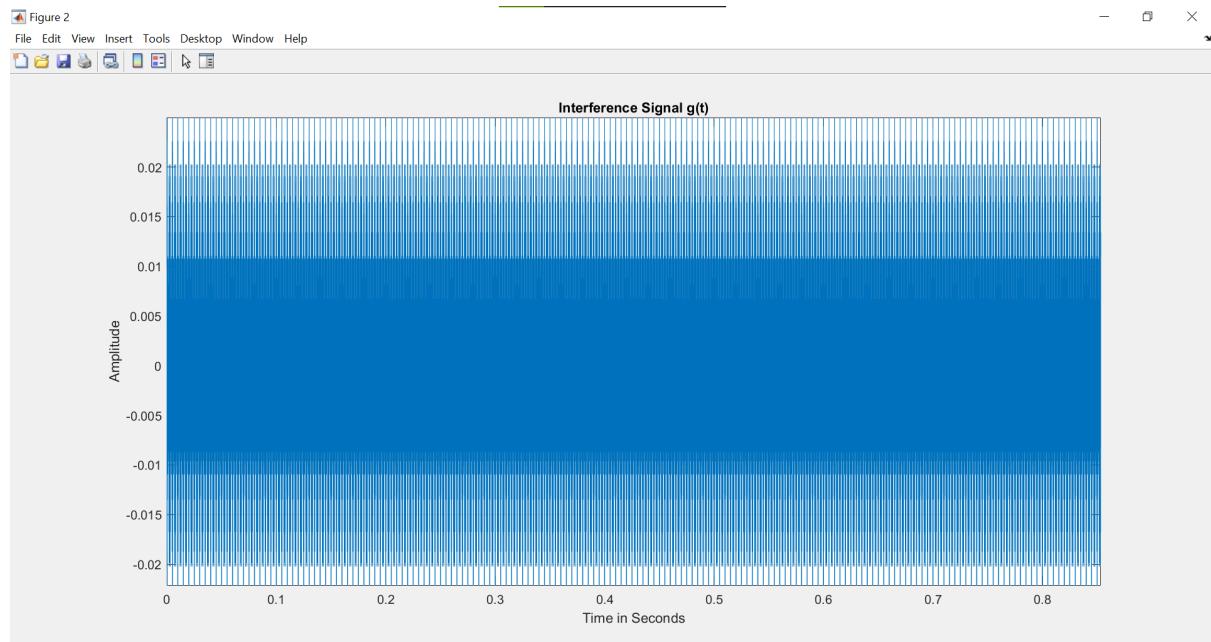


**Figure 22:** Magnitude of Spectrum of Received Signal  $X_r(f)$

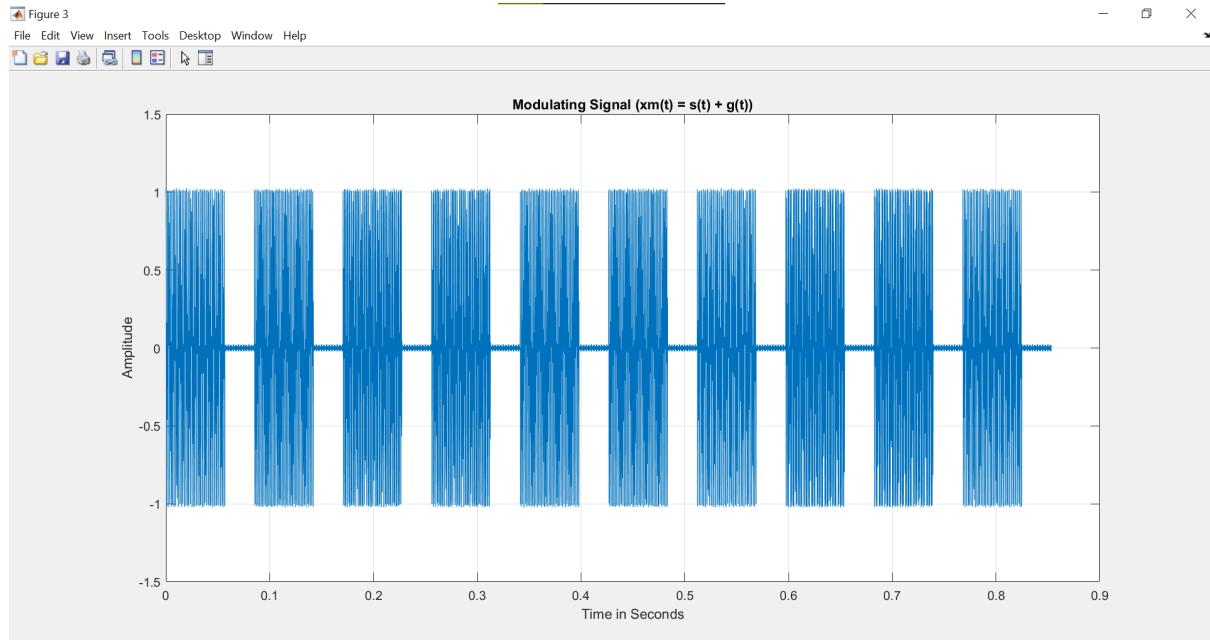
## Tasks 6:



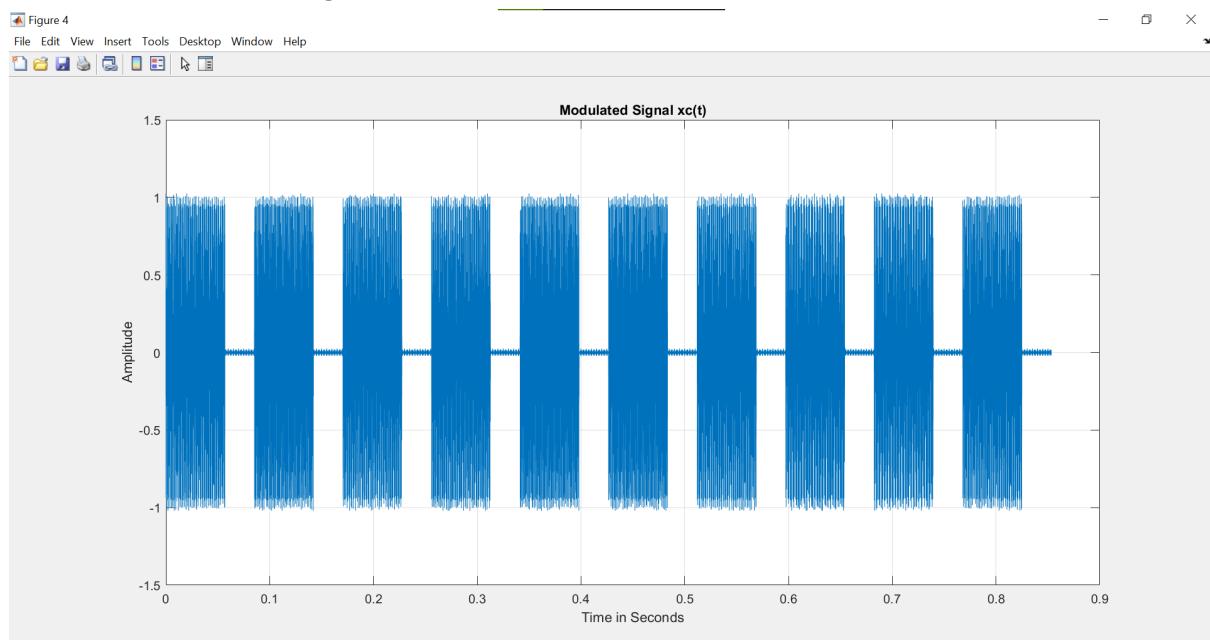
**Figure 23:** Wanted Signal  $s(t)$  in the Time Domain



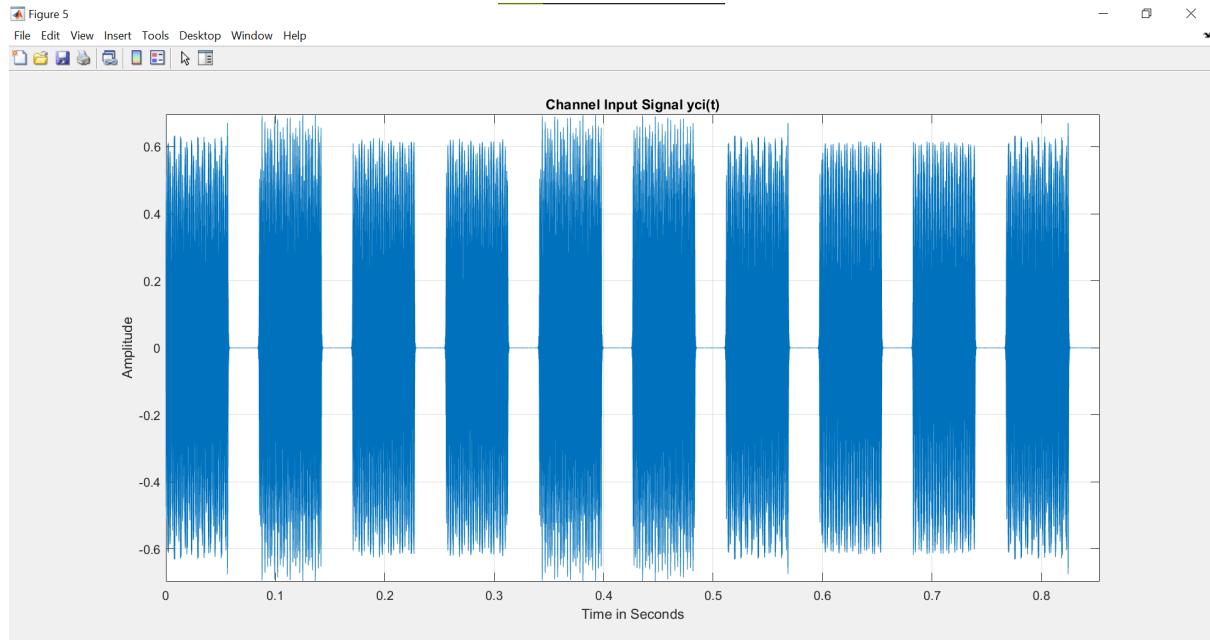
**Figure 24:** Interference Signal  $g(t)$  in the Time Domain



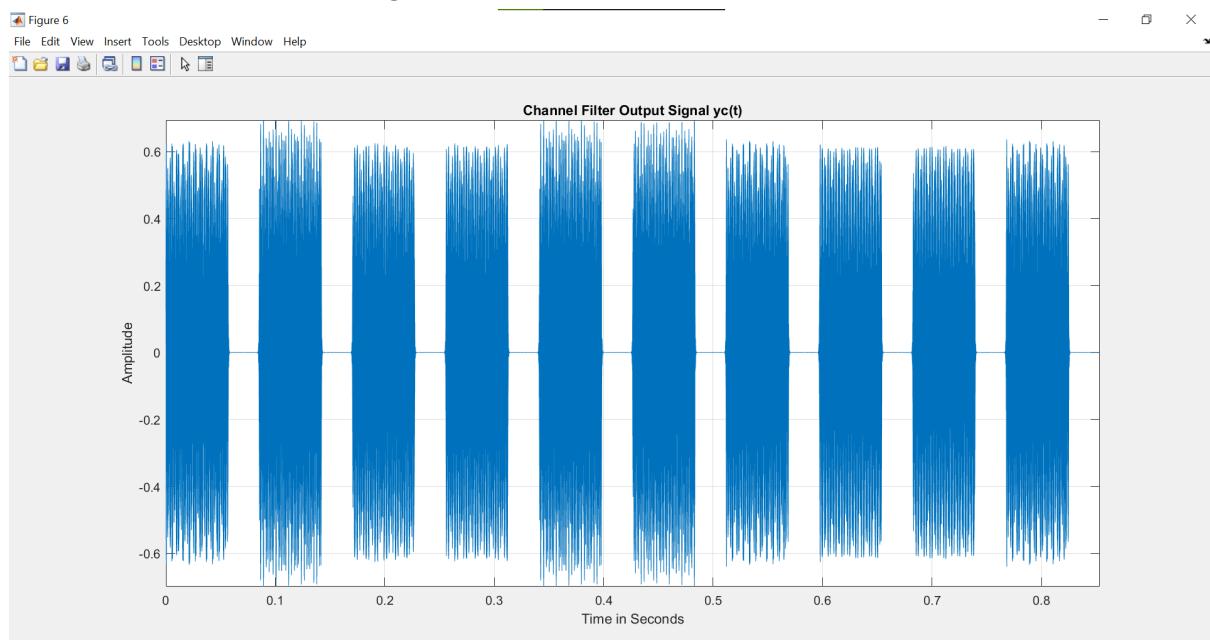
**Figure 25:** Modulating Signal ( $xm(t) = s(t) * g(t)$ )



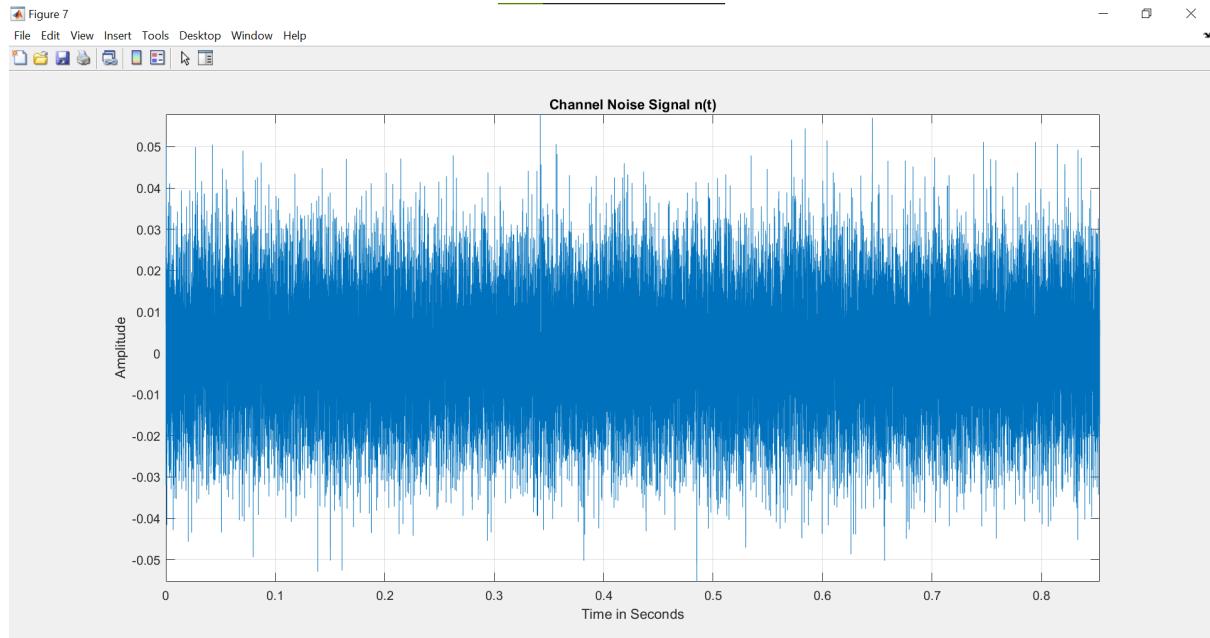
**Figure 26:** Modulated Signal  $xc(t)$



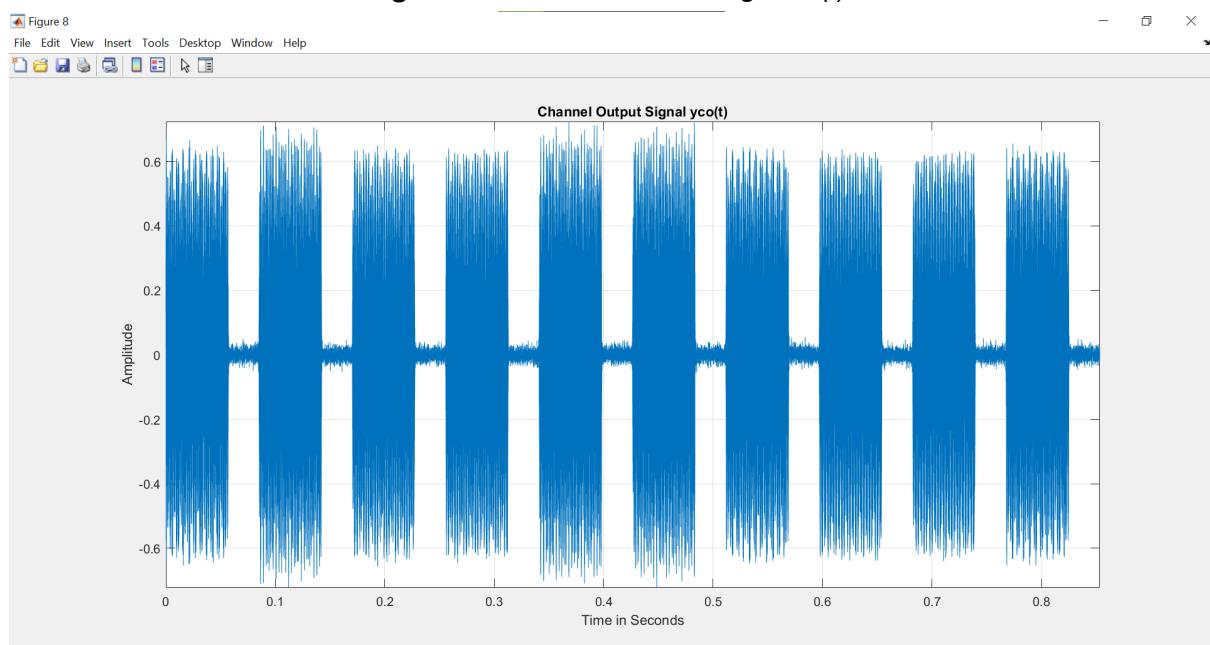
**Figure 27:** Channel Input Signal  $y_{ci}(t)$



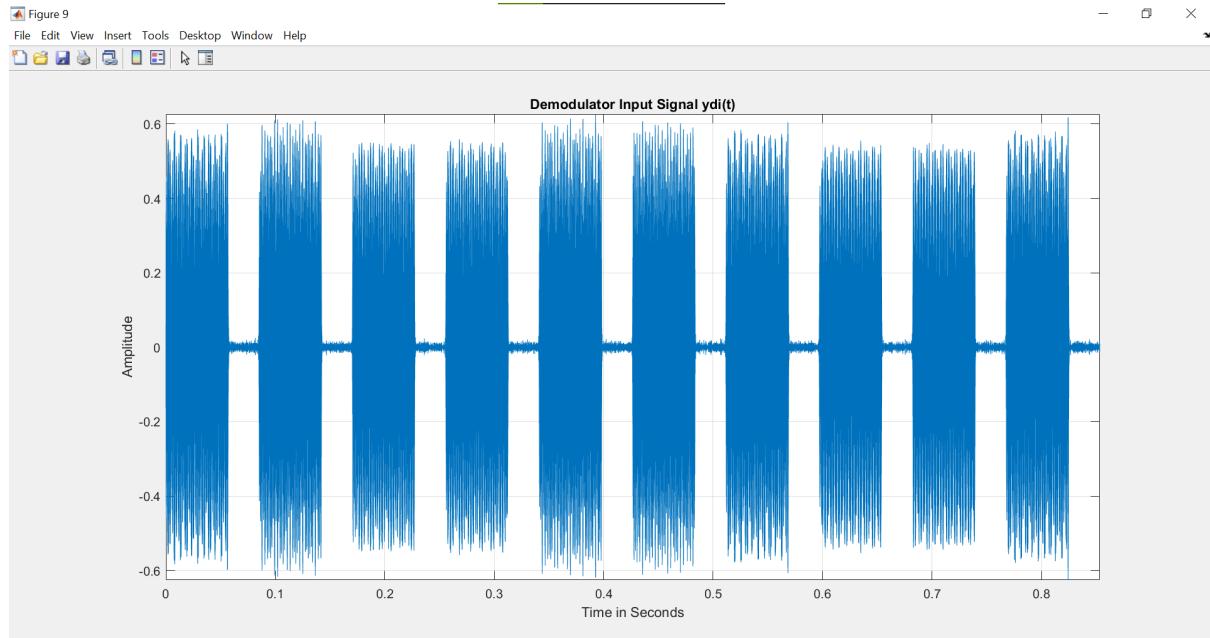
**Figure 28:** Channel Filter Output Signal  $y_{ci}(t)$



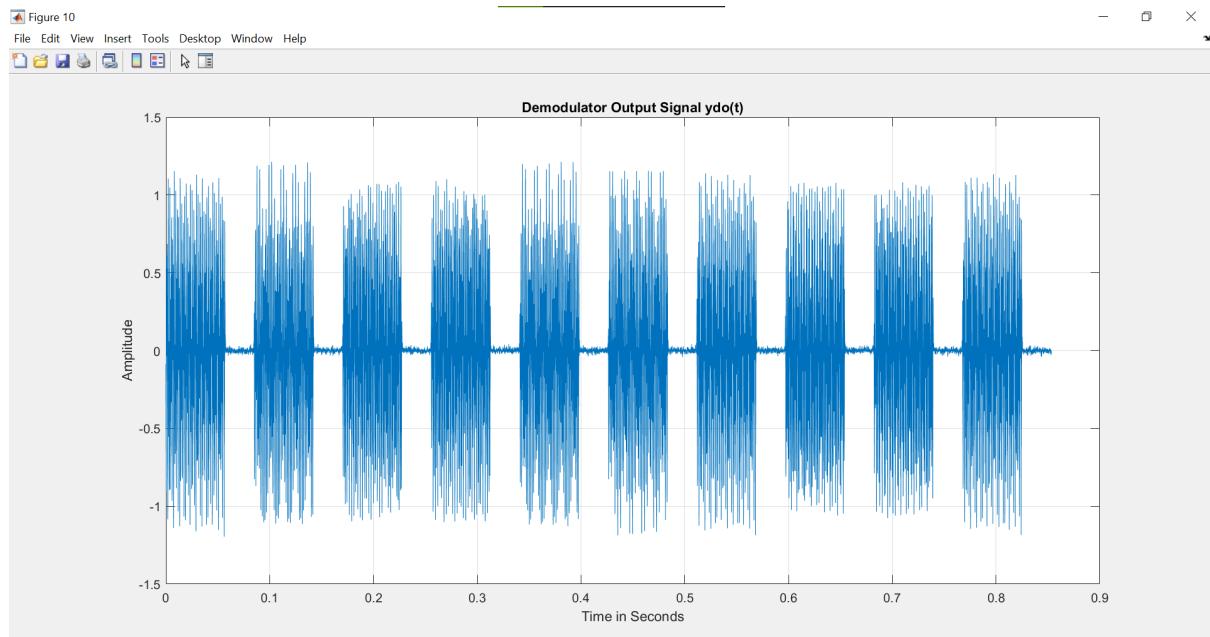
**Figure 29:** Channel Noise Signal  $n(t)$



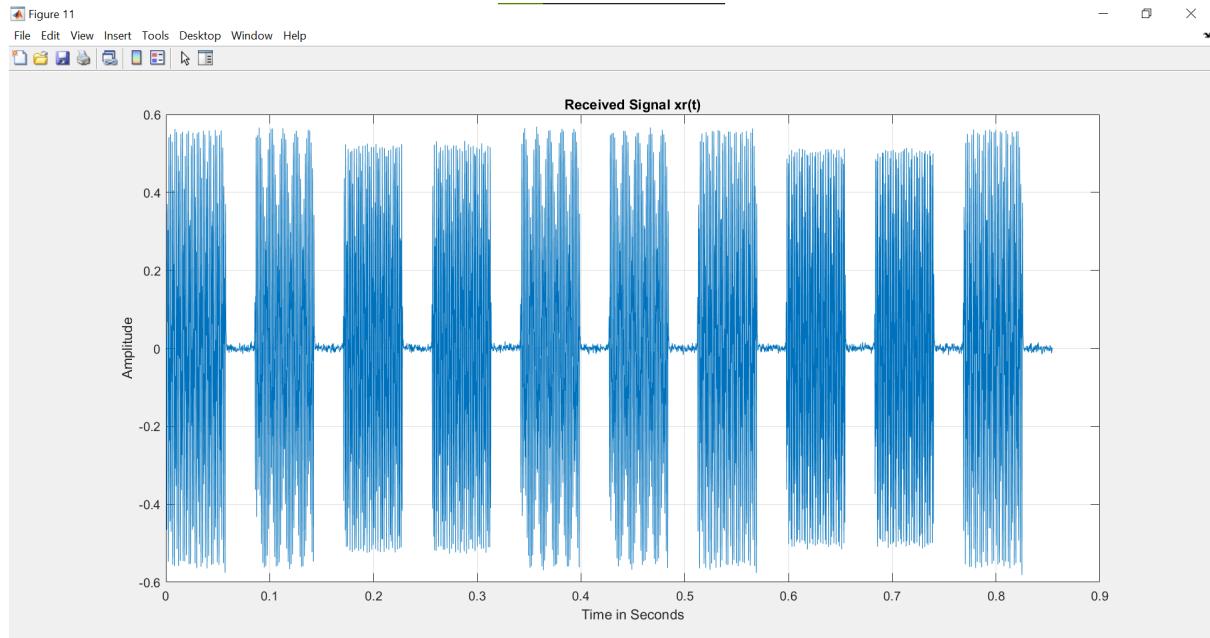
**Figure 30:** Channel Output Signal  $y_{co}(t)$



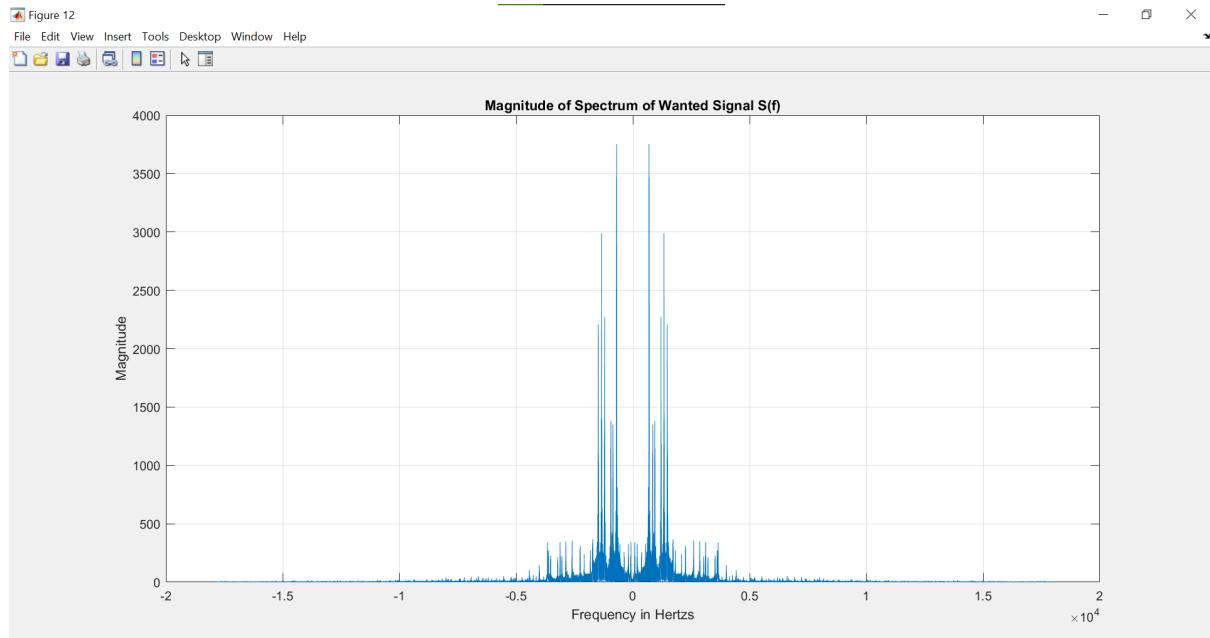
**Figure 31:** Demodulator Input Signal  $y_{di}(t)$



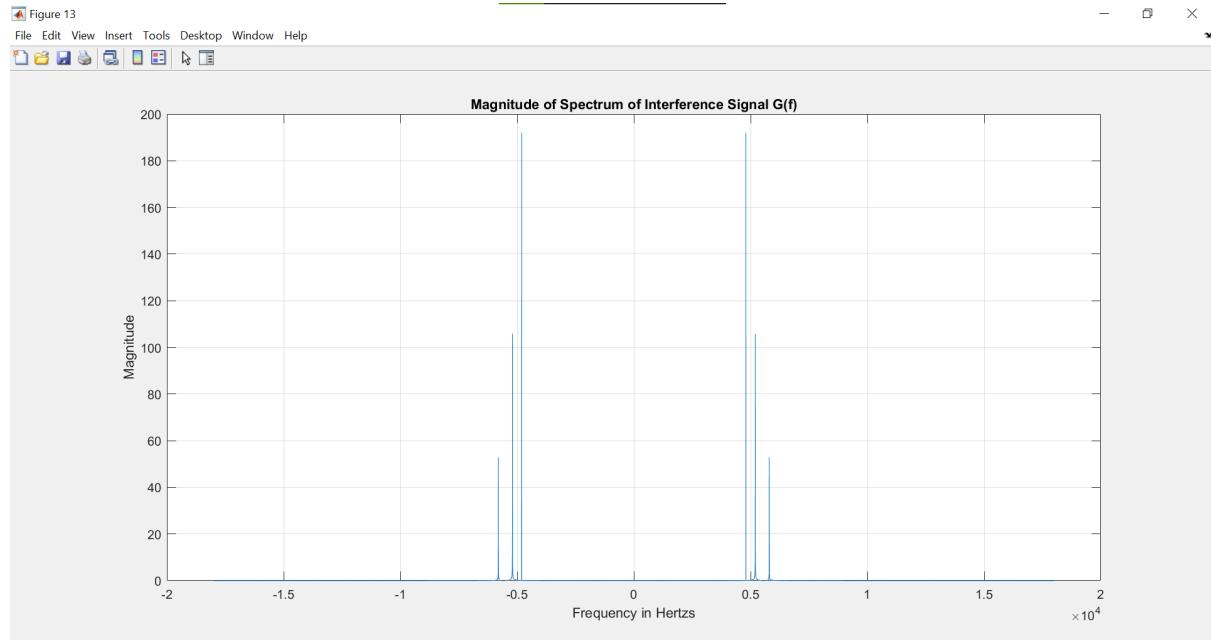
**Figure 32:** Demodulator Output Signal  $y_{do}(t)$



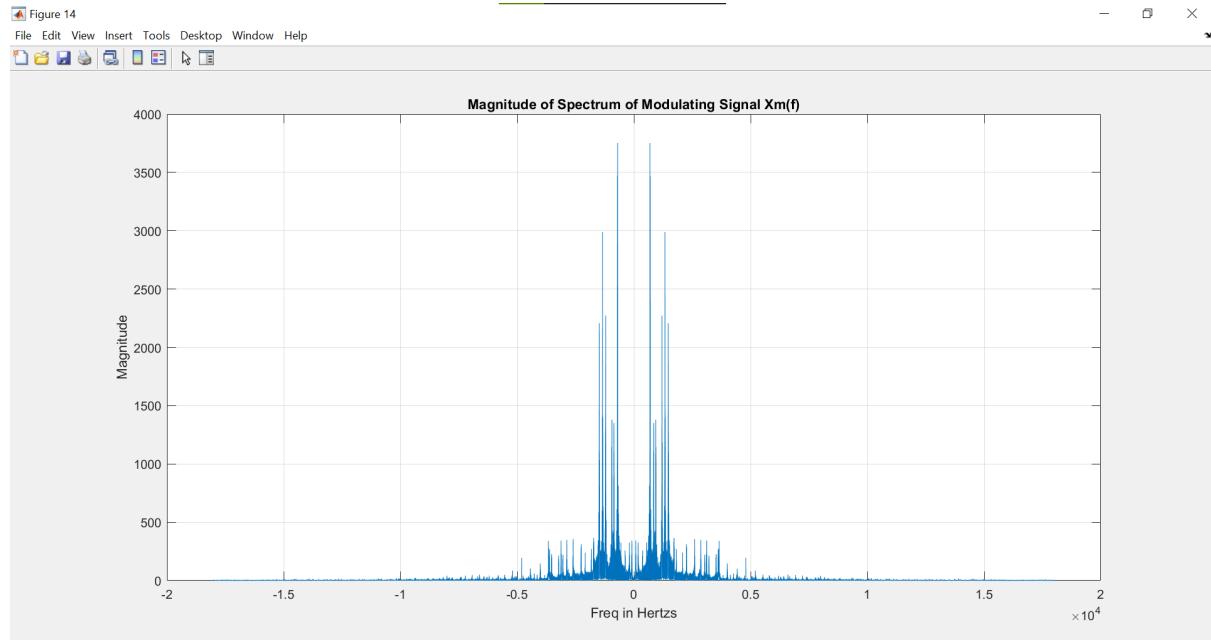
**Figure 33: Received Signal  $x_r(t)$**



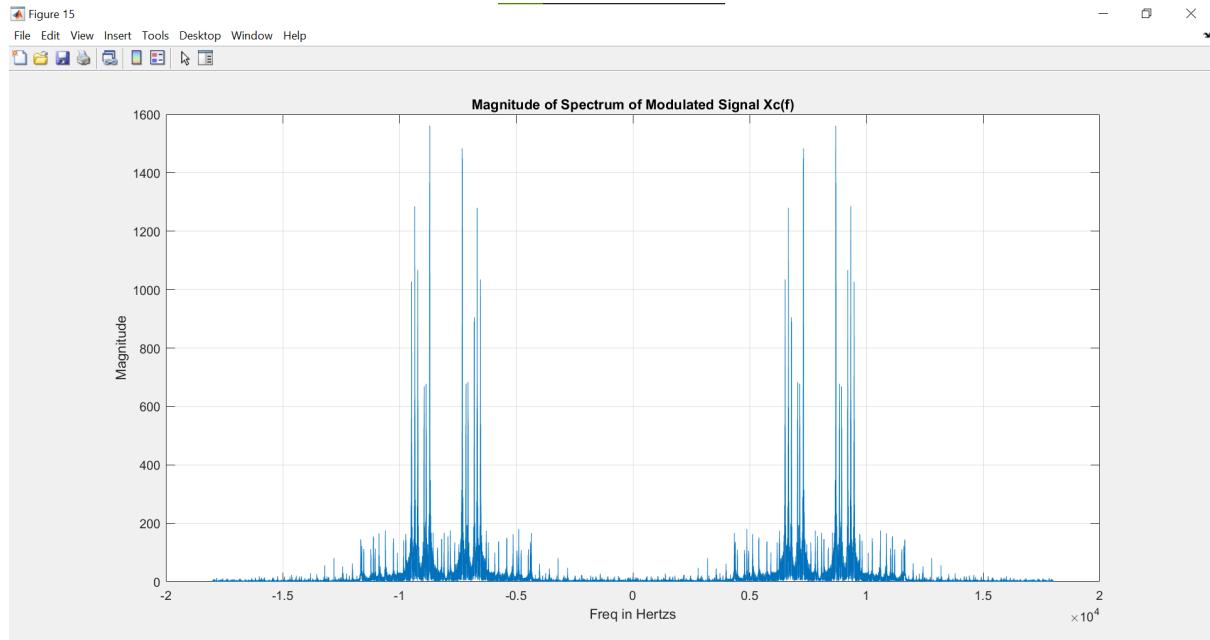
**Figure 34: Magnitude of Spectrum of Wanted Signal  $S(f)$**



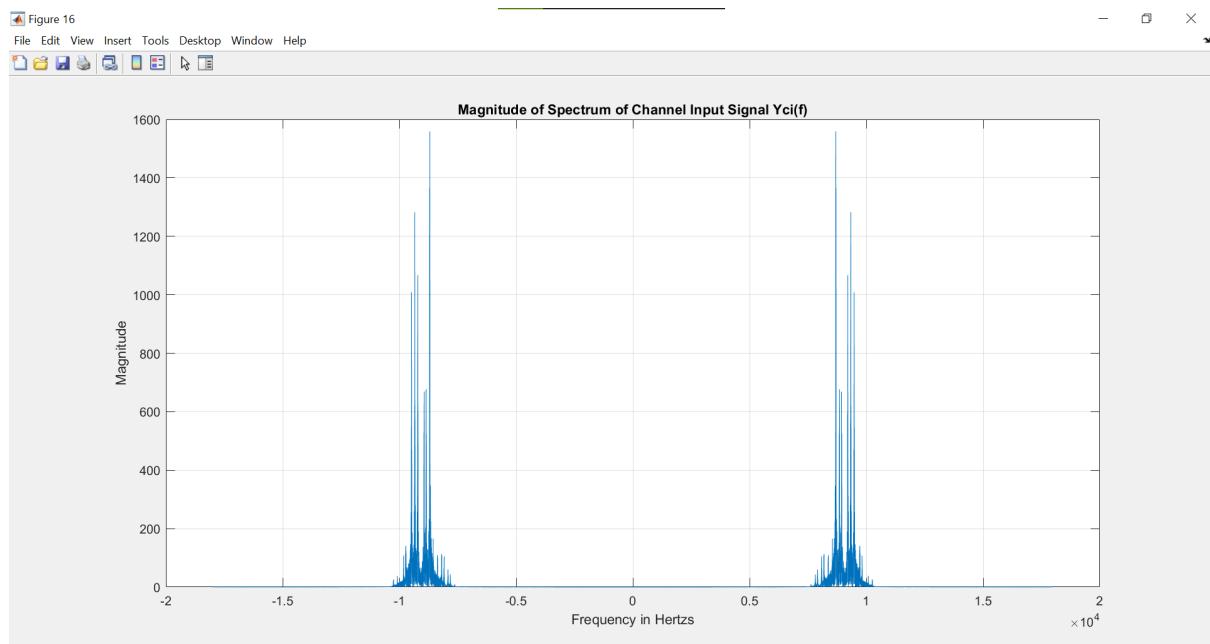
**Figure 35:** Magnitude of Spectrum of Interference Signal  $G(f)$



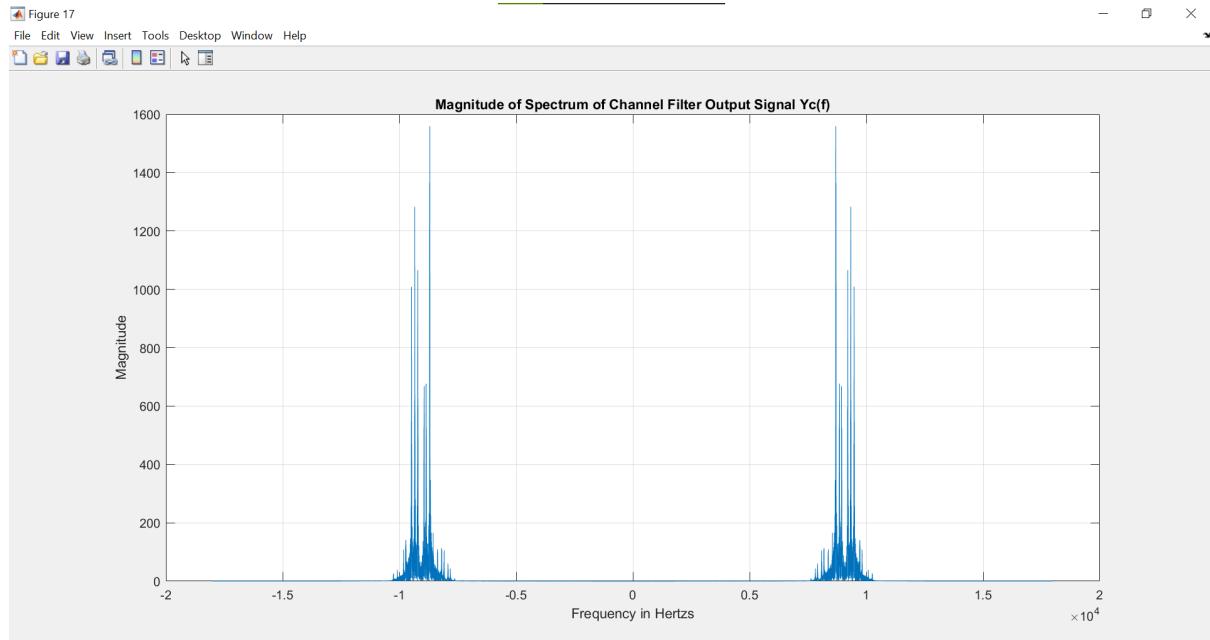
**Figure 36:** Magnitude of Spectrum of Modulating Signal  $Xm(f)$



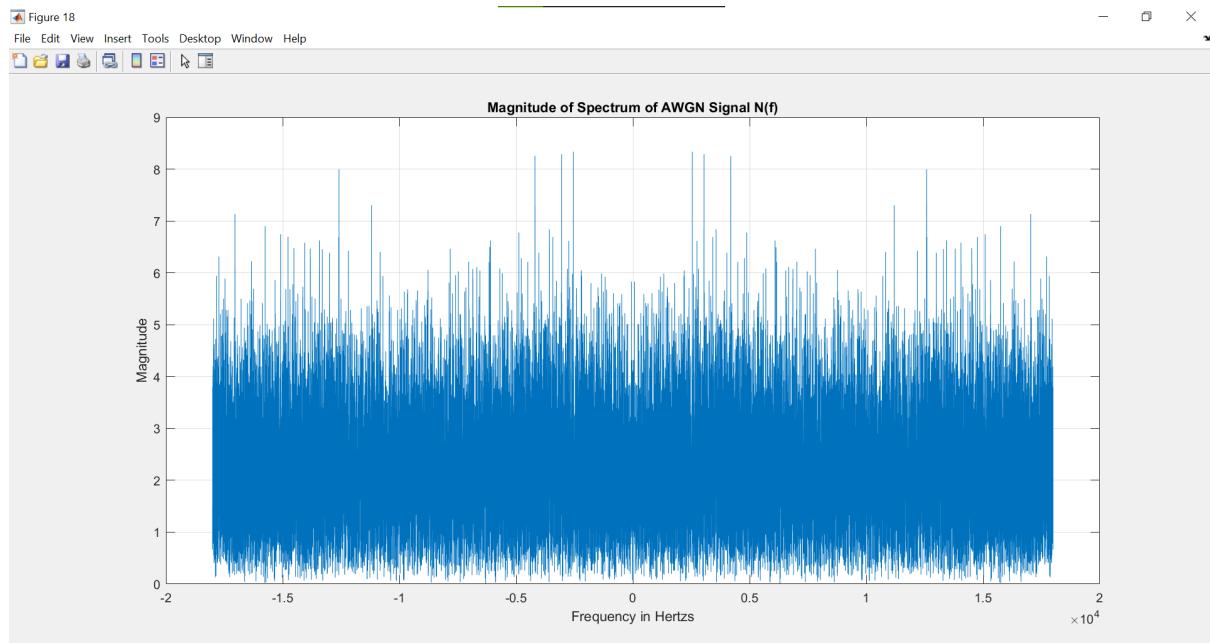
**Figure 37:** Magnitude of Spectrum of Modulated Signal  $X_c(f)$



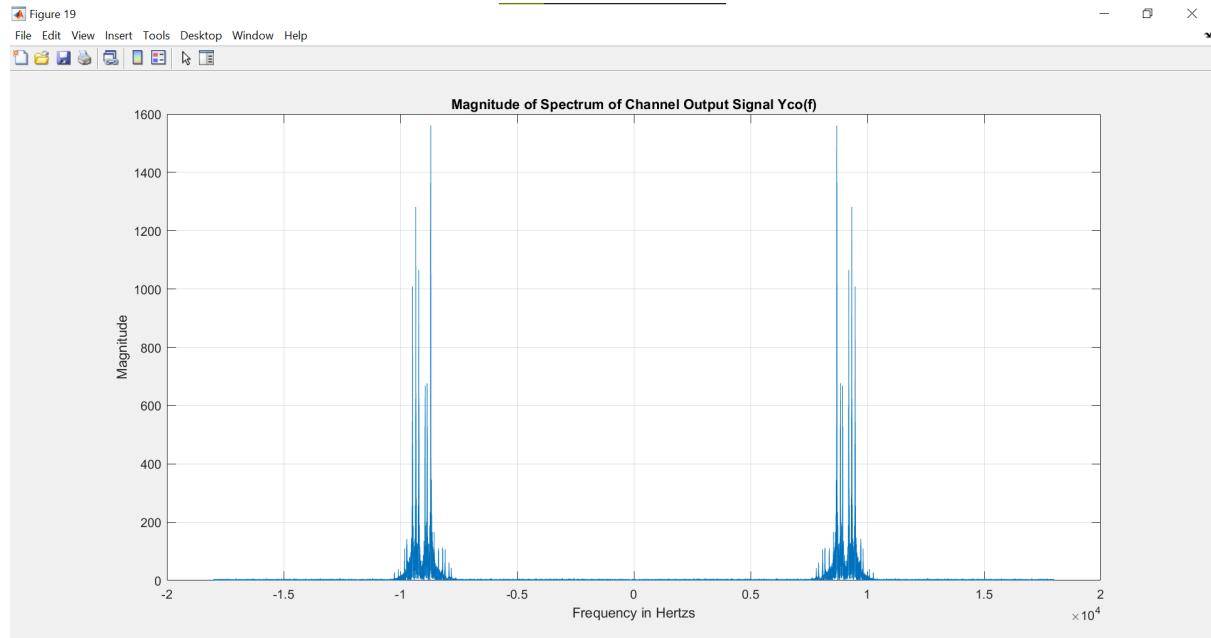
**Figure 38:** Magnitude of Spectrum of Channel Input Signal  $Y_{ci}(f)$



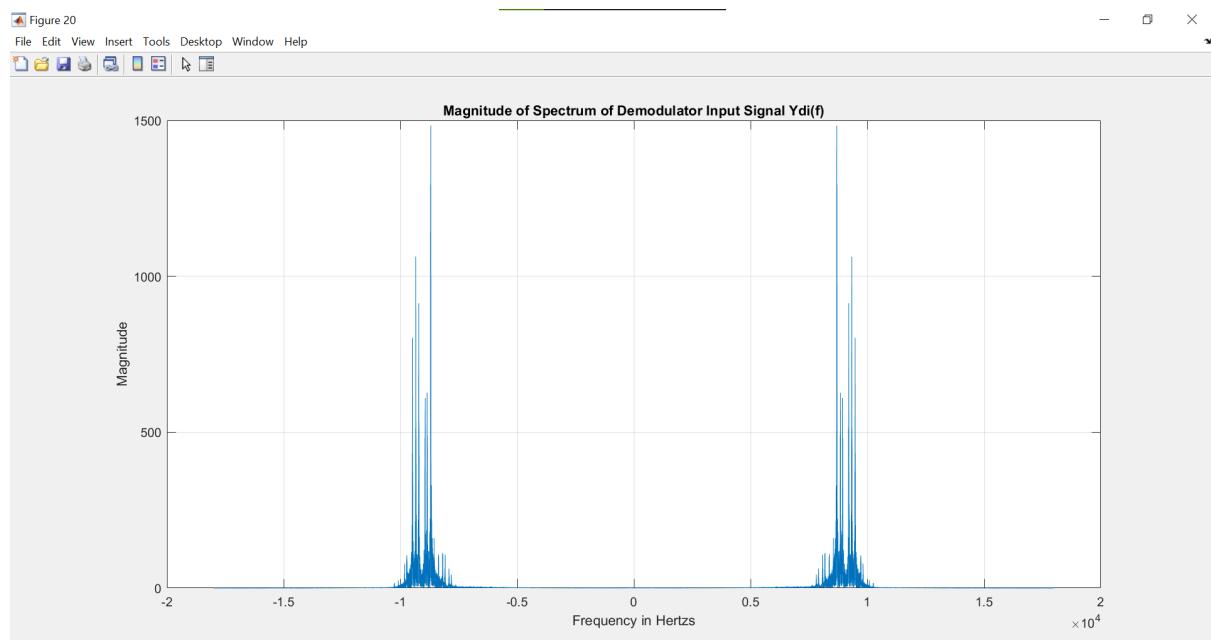
**Figure 39:** Magnitude of Spectrum of Channel Filter Output Signal  $Y_c(f)$



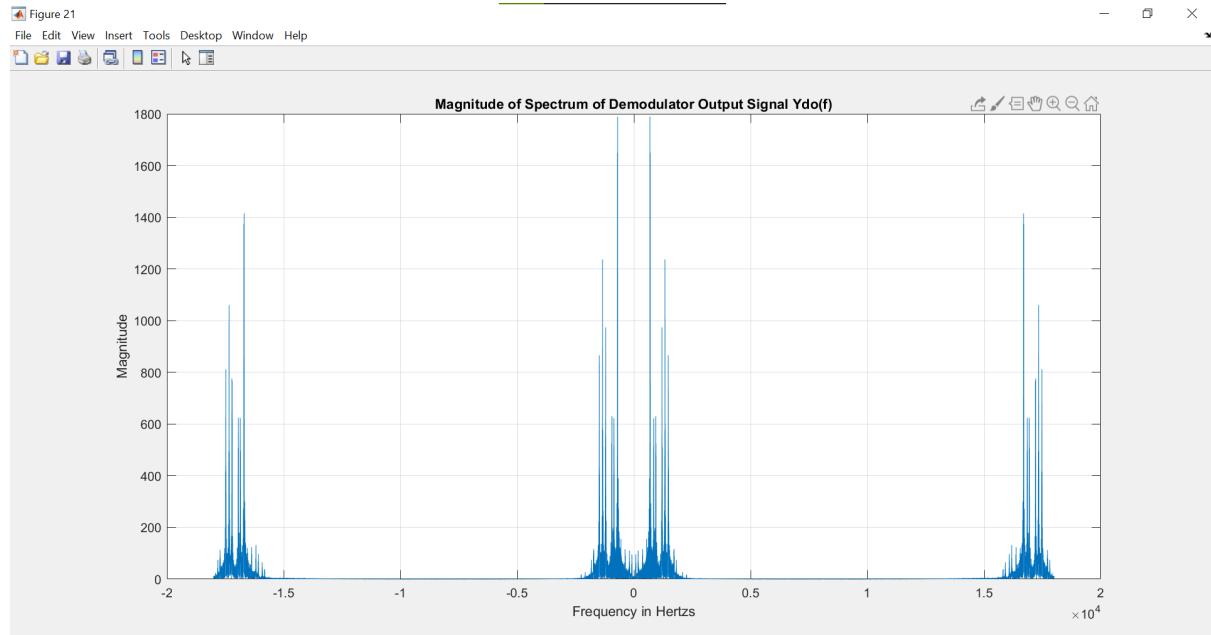
**Figure 40:** Magnitude of Spectrum AWGN Signal  $N(f)$



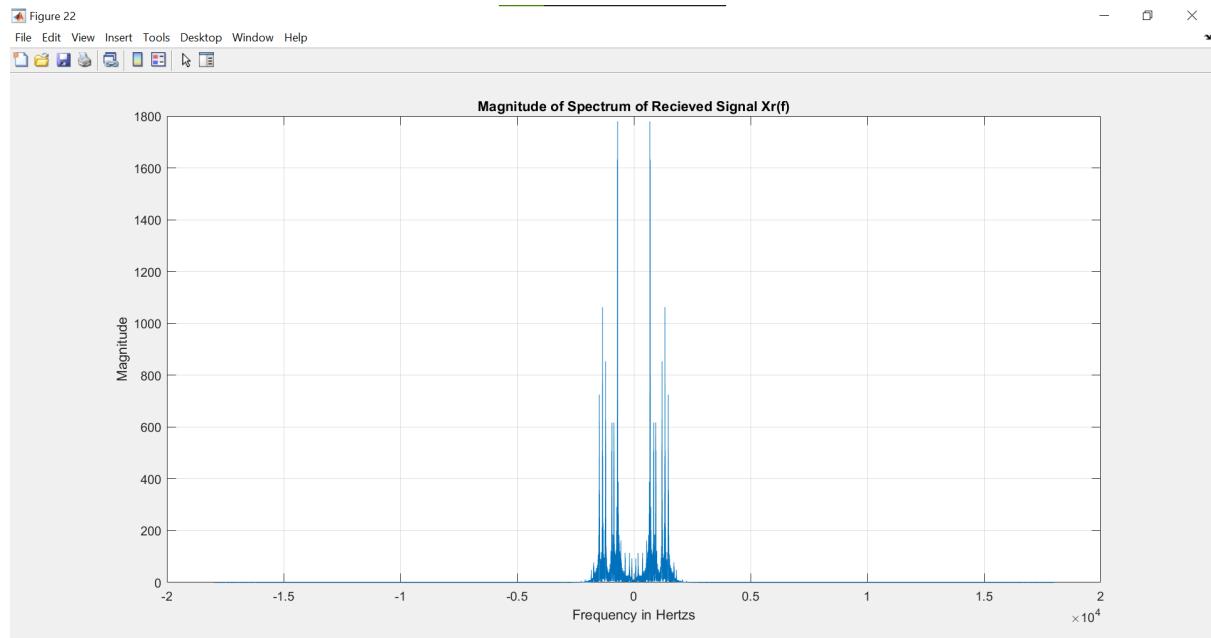
**Figure 41:** Magnitude of Spectrum of Channel Output Signal  $Y_{CO}(f)$



**Figure 42:** Magnitude of Spectrum of Demodulator Input Signal  $Y_{DI}(f)$



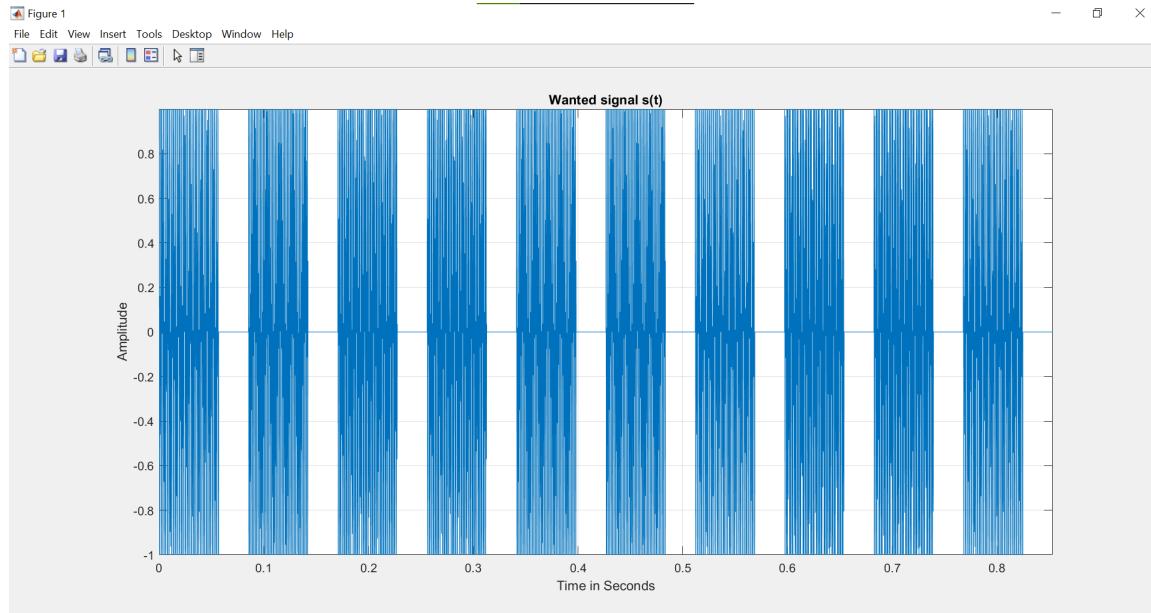
**Figure 43:** Magnitude of Spectrum of Demodulator Output Signal  $Y_{do}(f)$



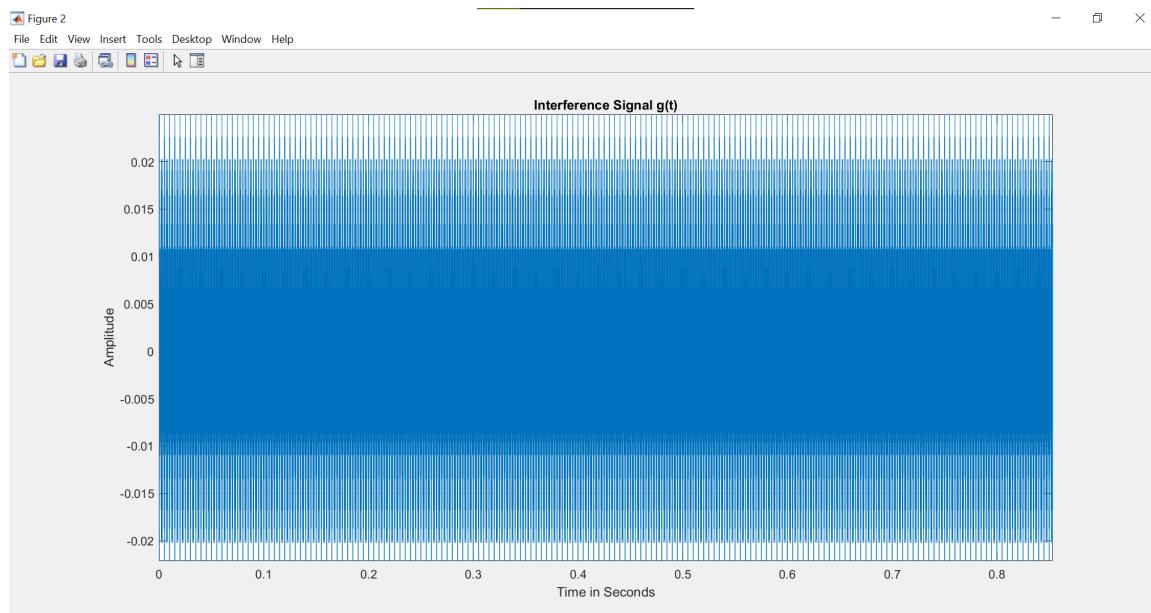
**Figure 44:** Magnitude of Spectrum of Received Signal  $X_r(f)$

## Task 2:

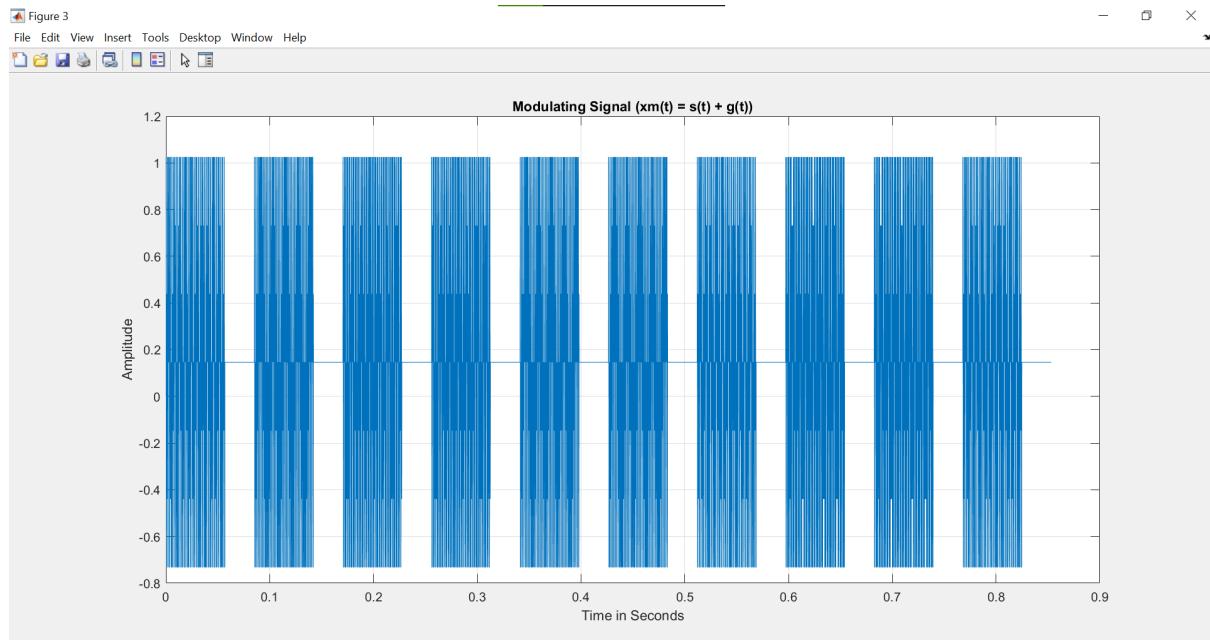
### 2.1



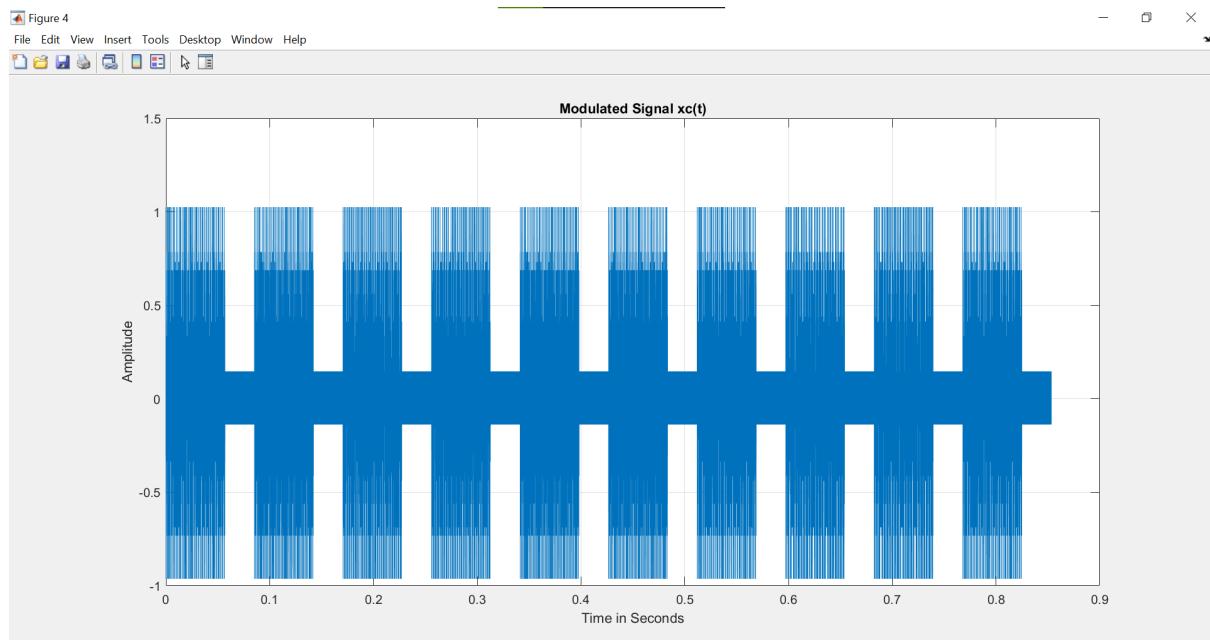
**Figure 45:** Wanted Signal in the Time Domain



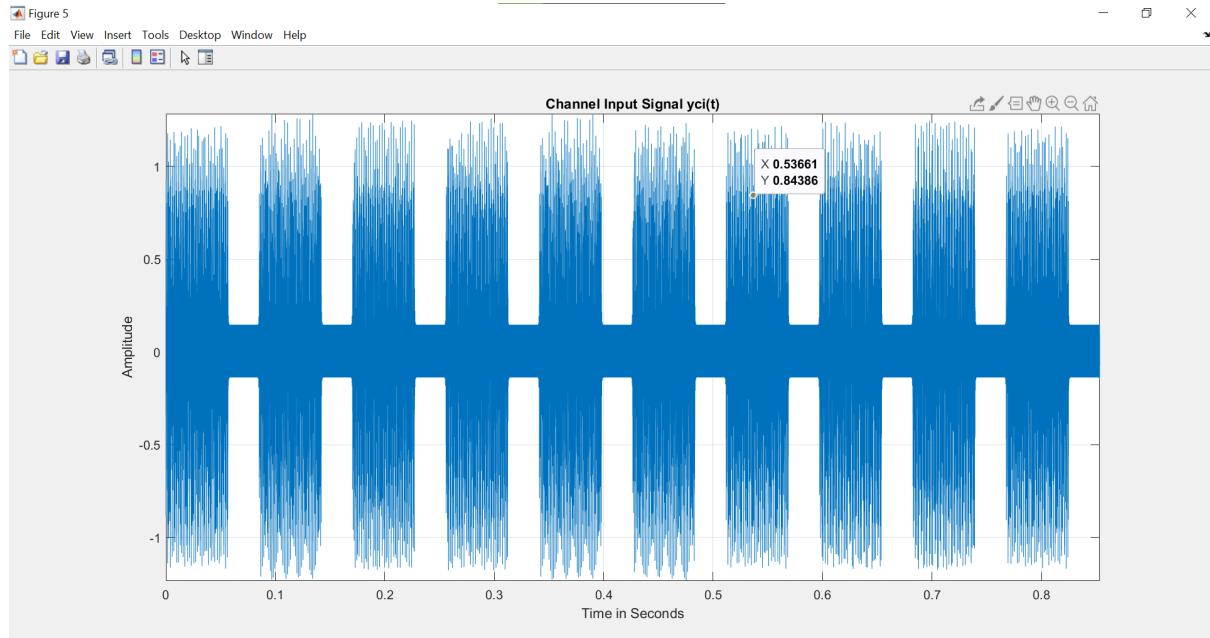
**Figure 46:** Interference Signal in the Time Domain



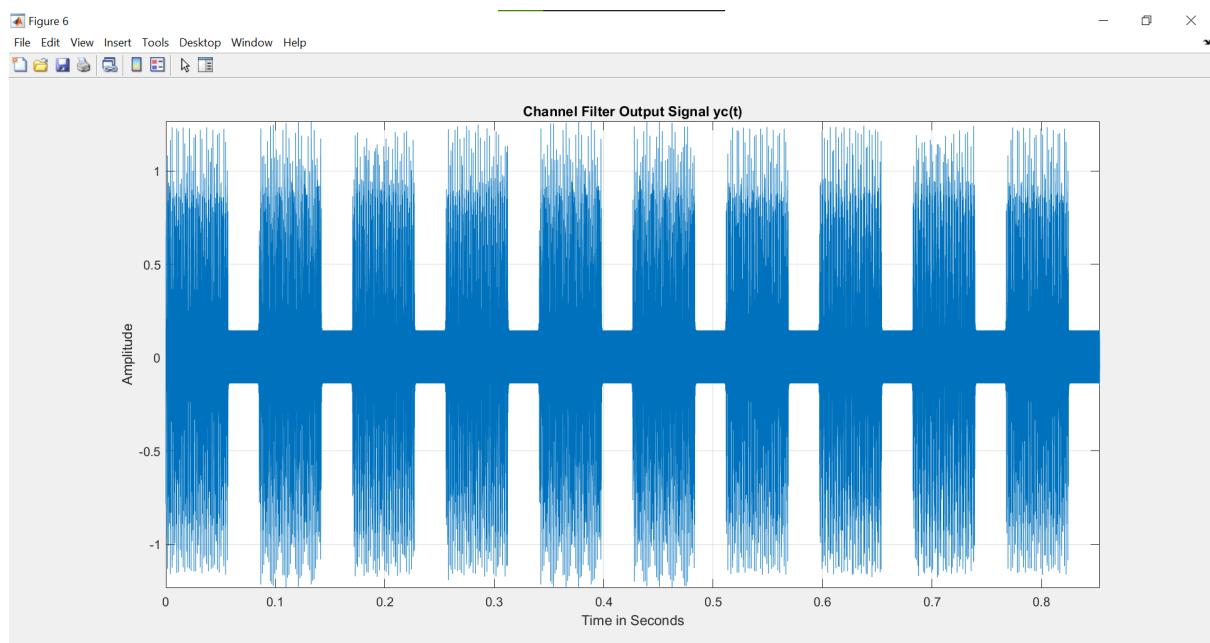
**Figure 47:** Modulating Signal ( $xm(t) = s(t) * g(t)$ )



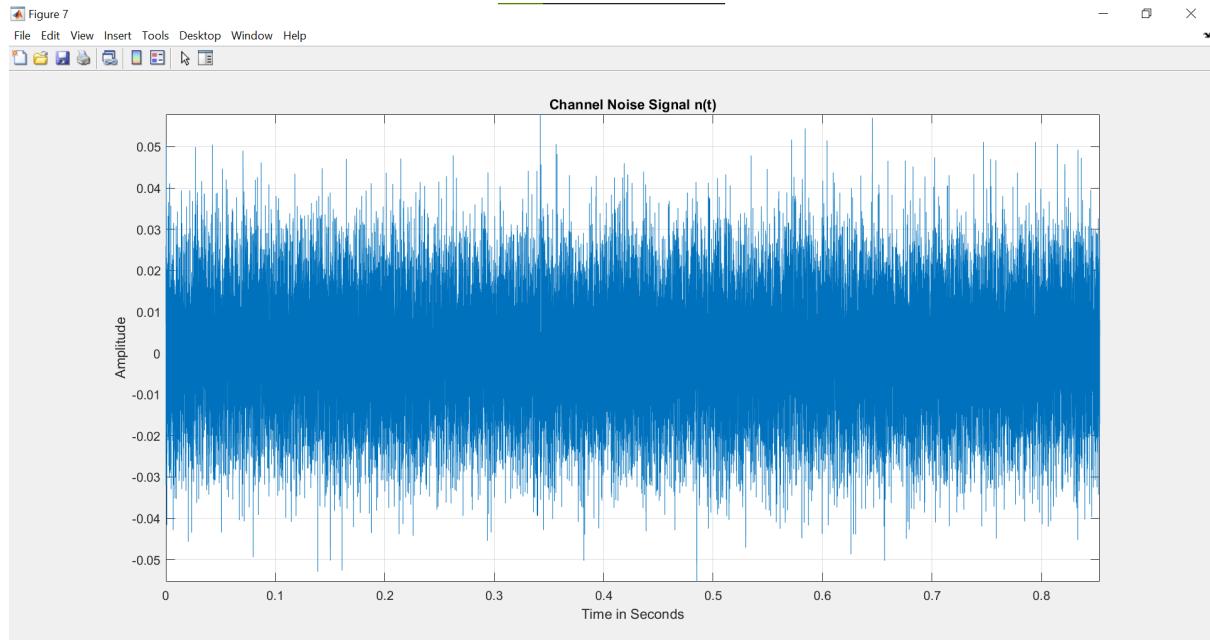
**Figure 48:** Modulated Signal  $xc(t)$



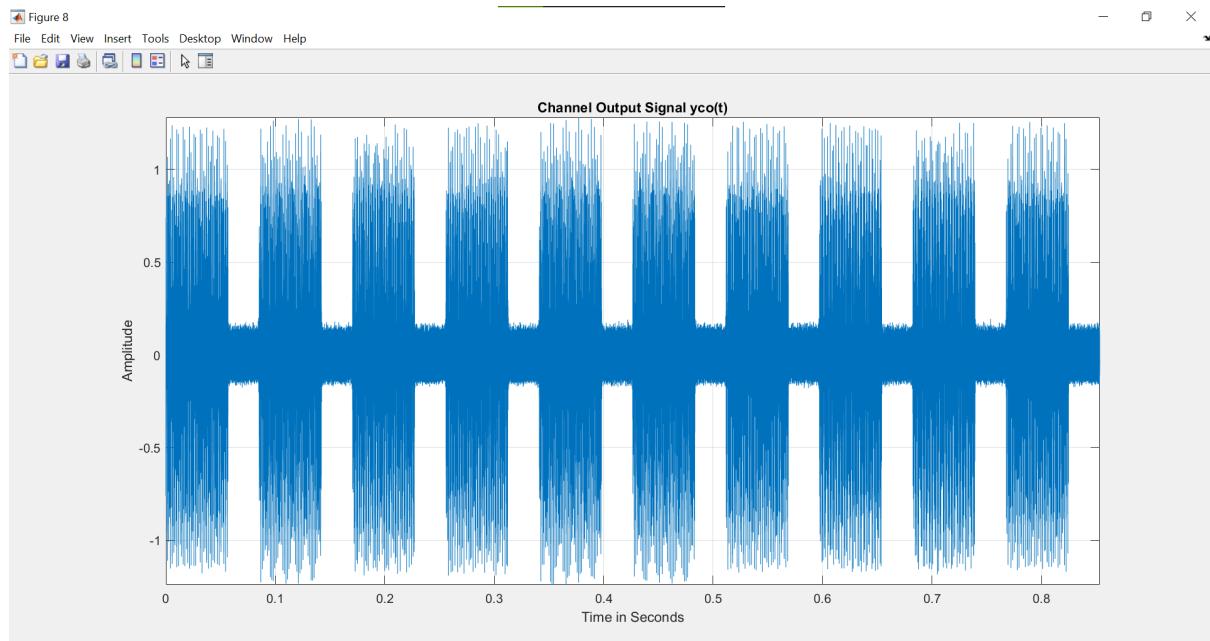
**Figure 49:** Channel Input Signal  $y_{ci}(t)$



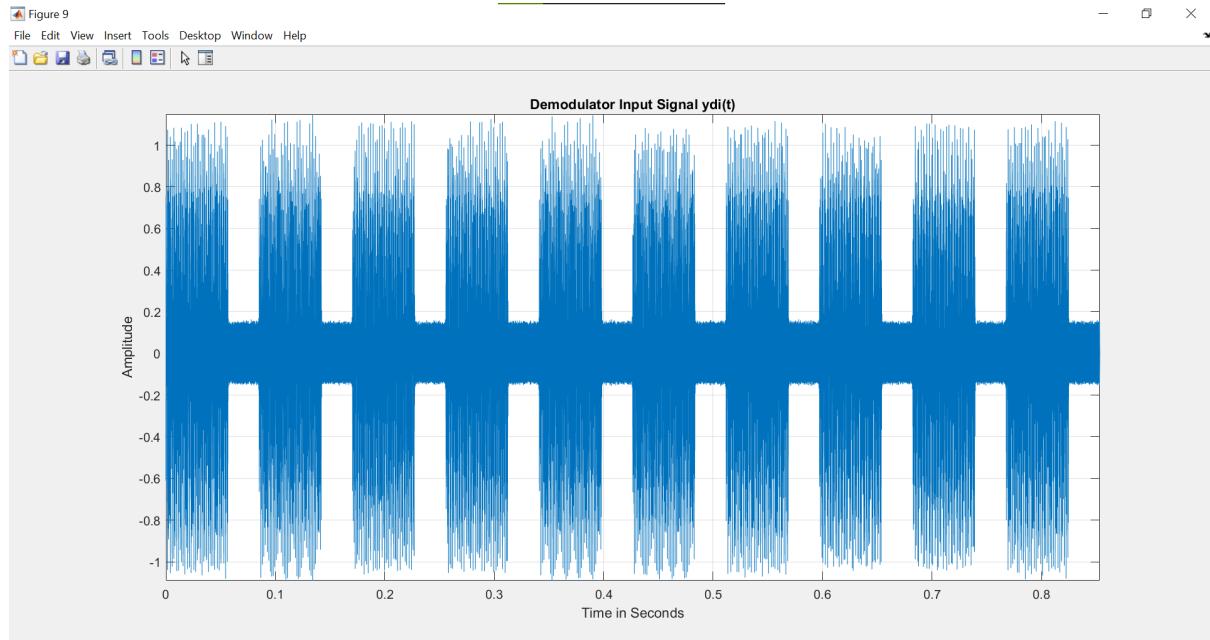
**Figure 50:** Channel Filter Output Signal  $y_c(t)$



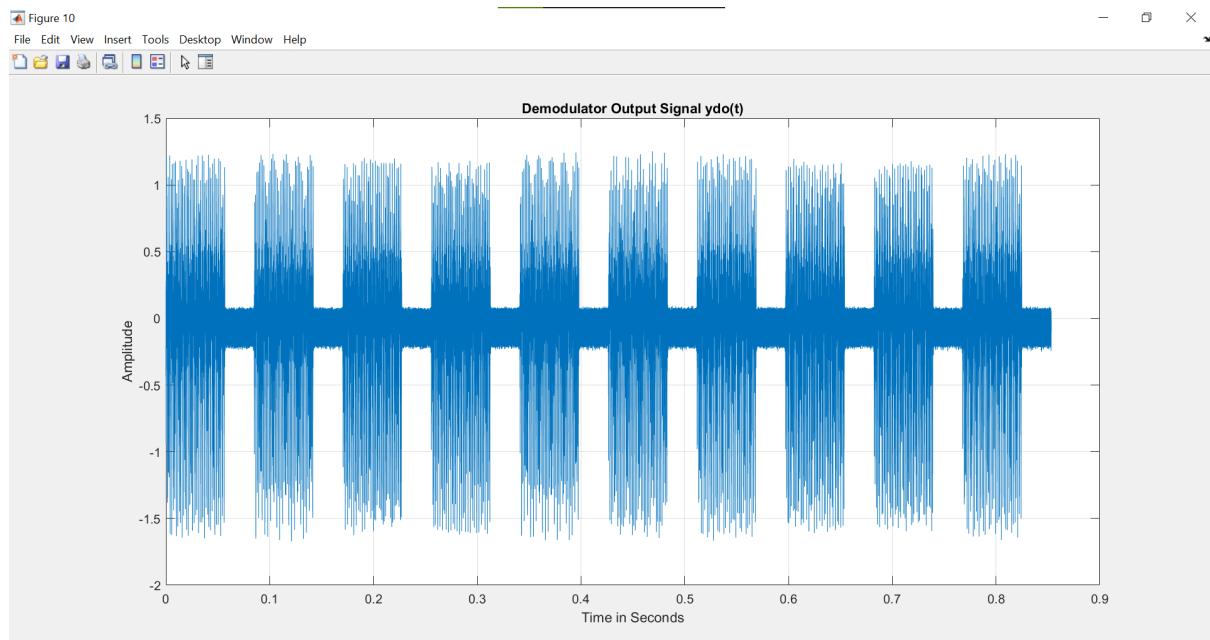
**Figure 51:** Channel Noise Signal  $n(t)$



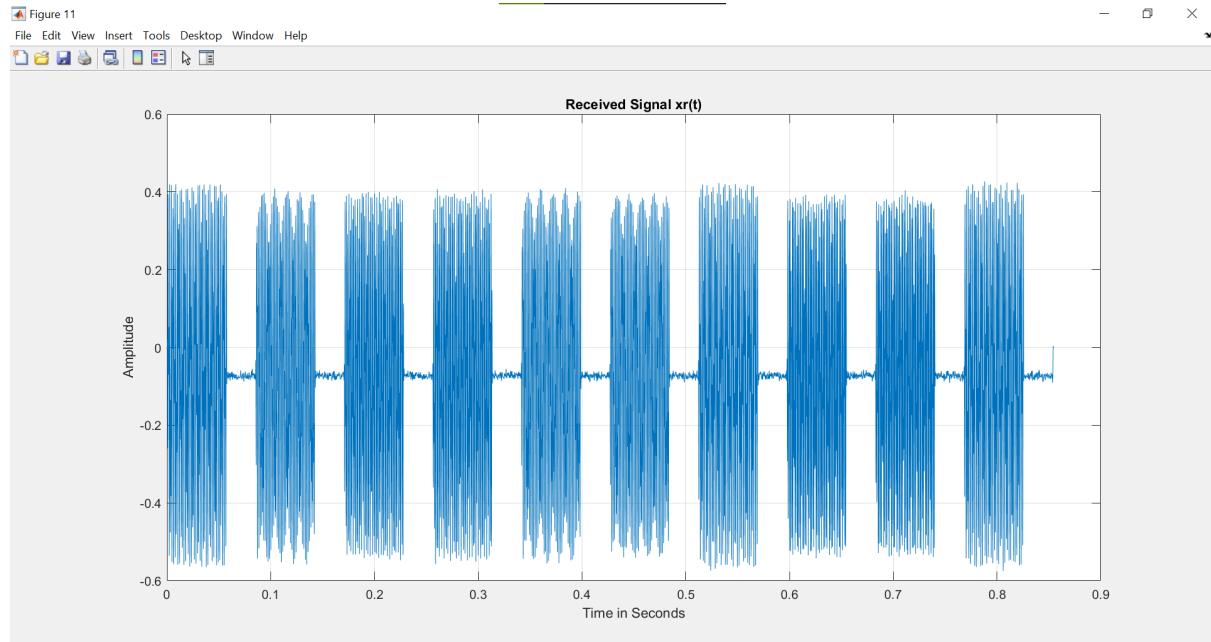
**Figure 52:** Channel Output Signal  $y_{co}(t)$



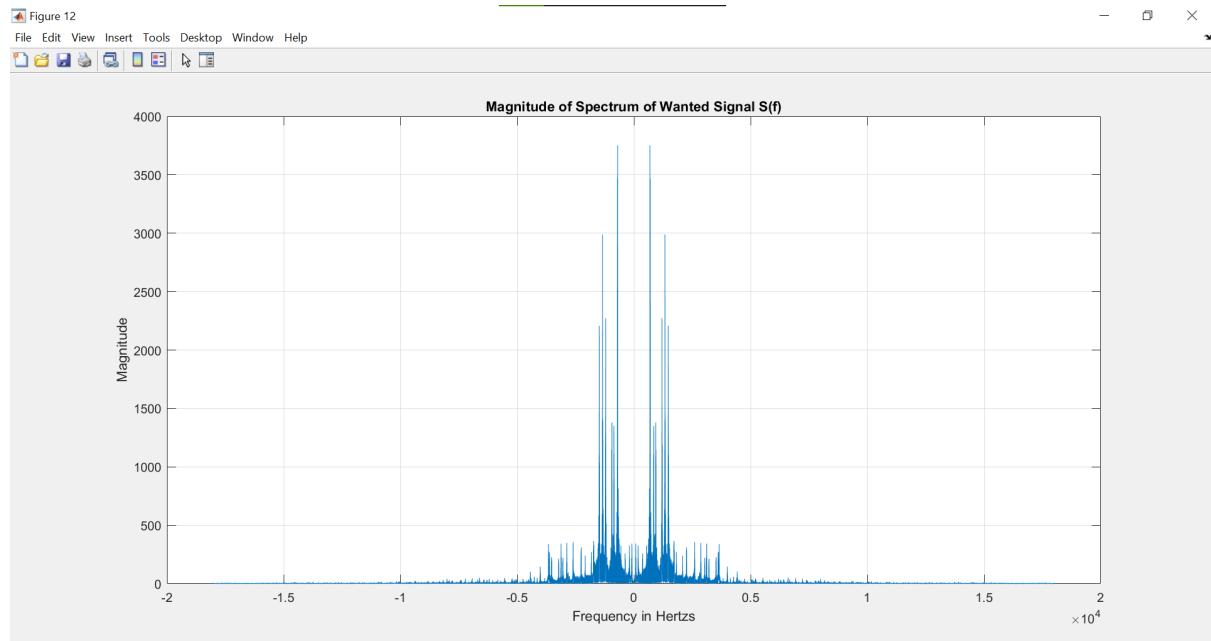
**Figure 53:** Demodulator Input Signal  $y_{di}(t)$



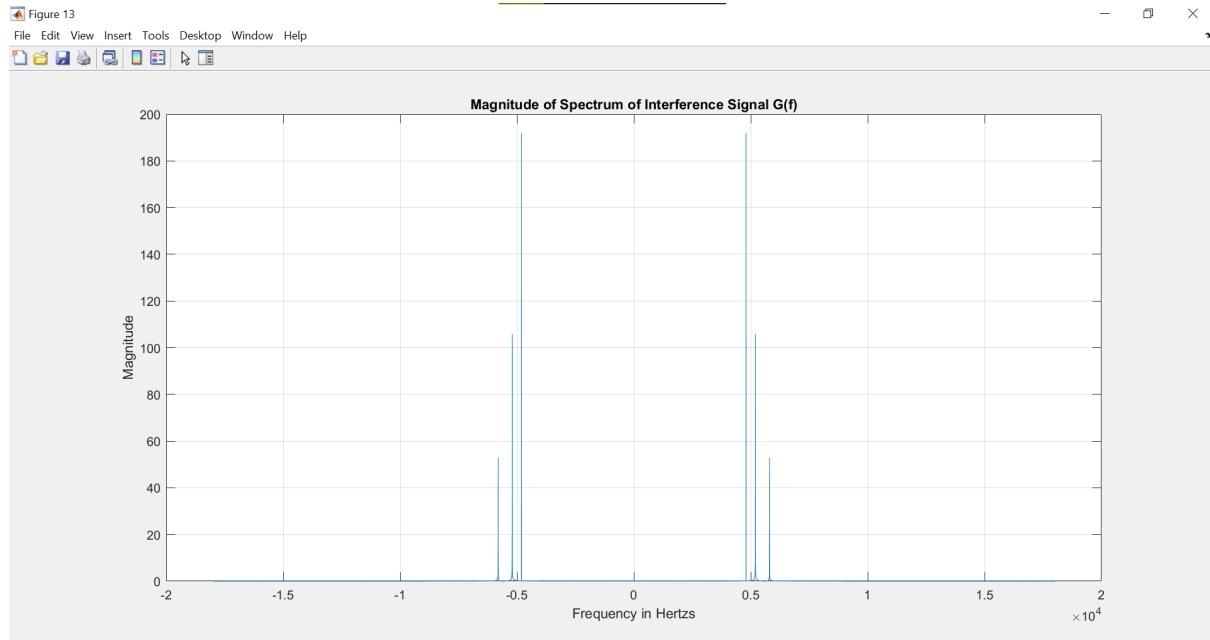
**Figure 54:** Demodulator Output Signal  $y_{do}(t)$



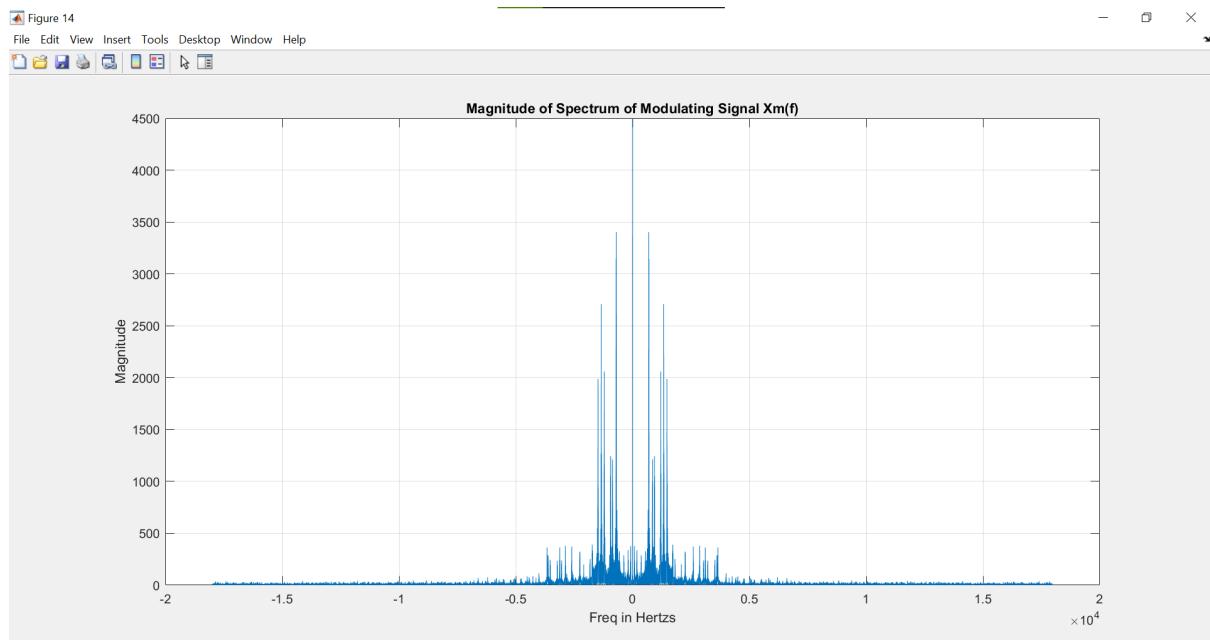
**Figure 55:** Received Signal  $x_r(t)$



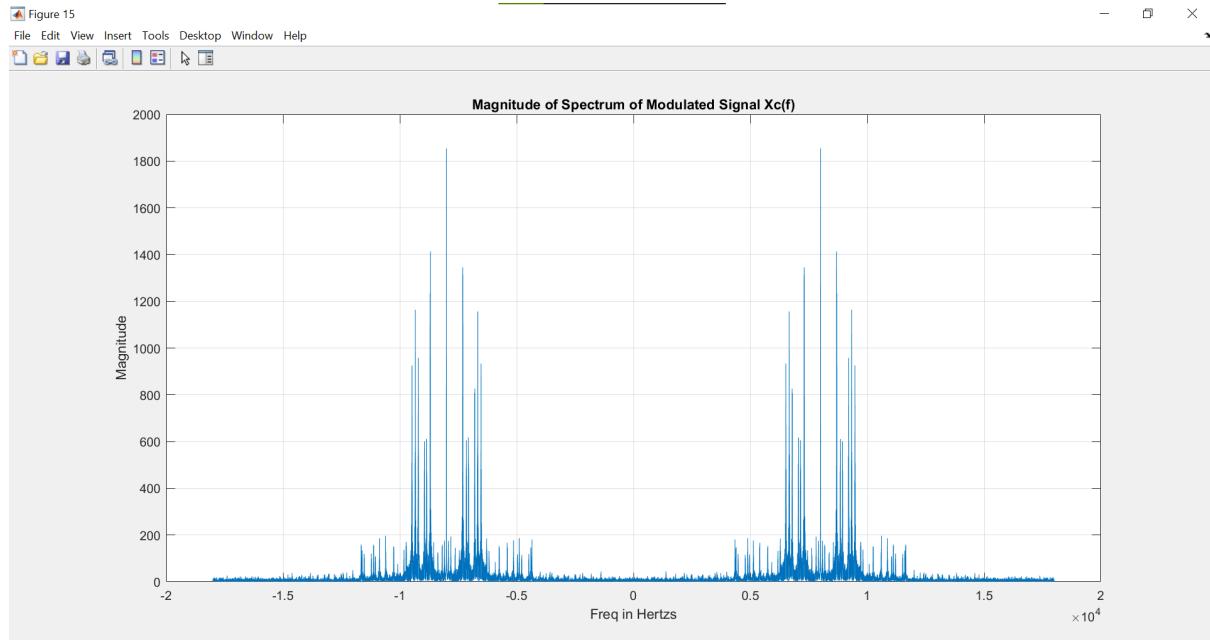
**Figure 56:** Magnitude of Spectrum of Wanted Signal  $S(f)$



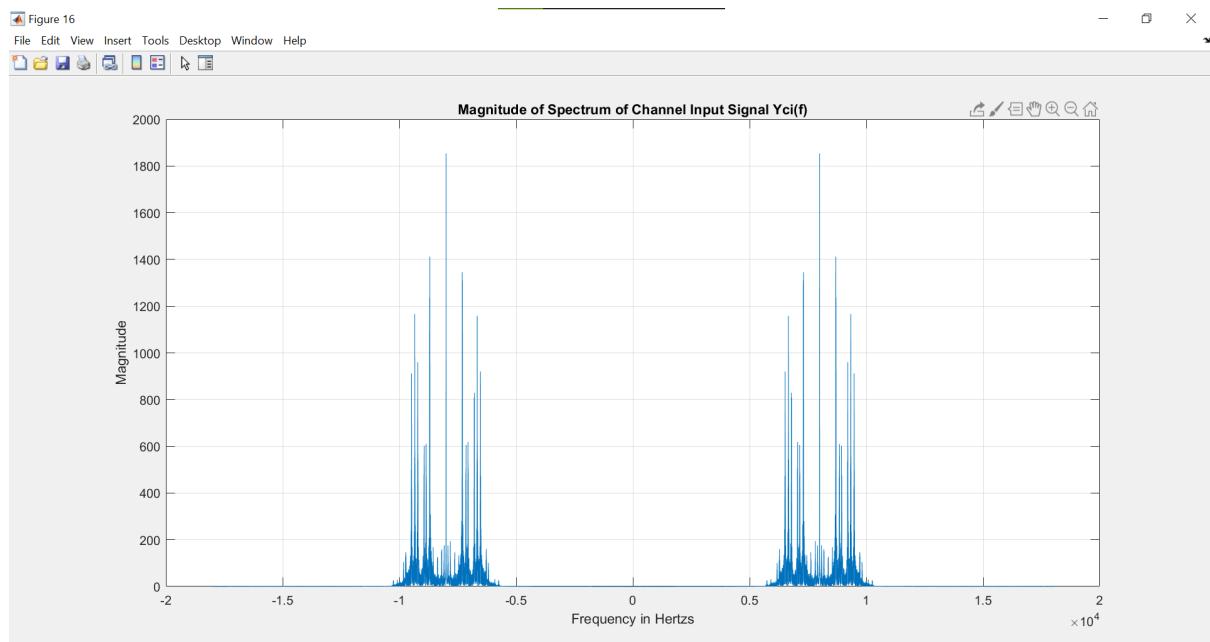
**Figure 57:** Magnitude of Spectrum of Interference Signal  $G(f)$



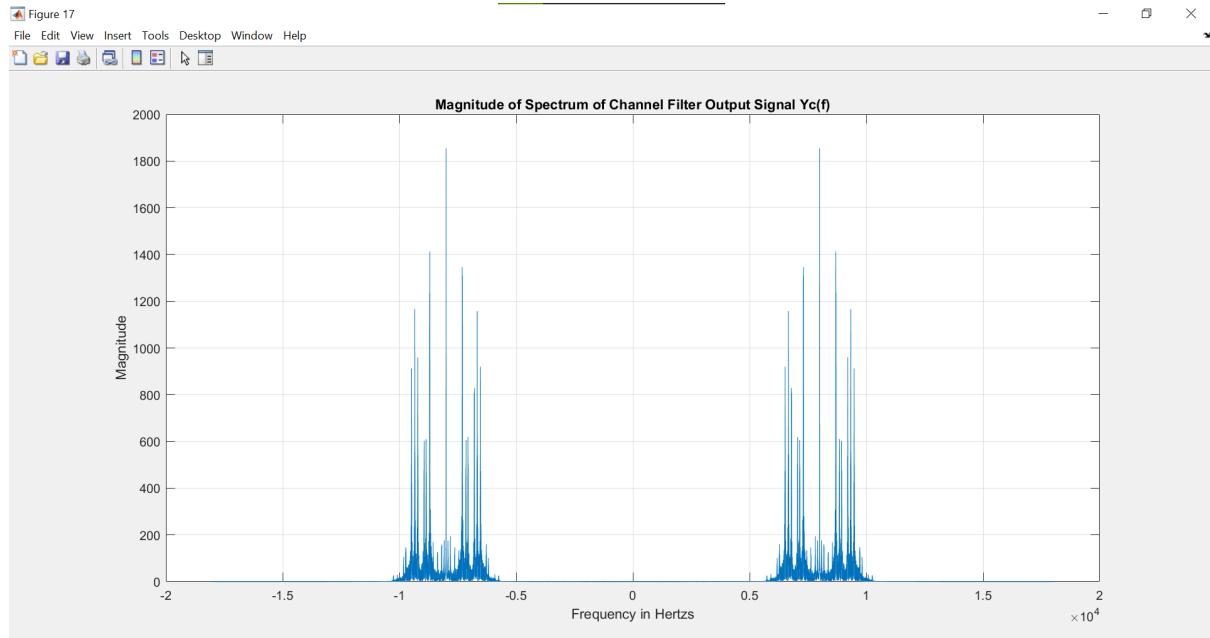
**Figure 58:** Magnitude of Spectrum of Modulating Signal  $Xm(f)$



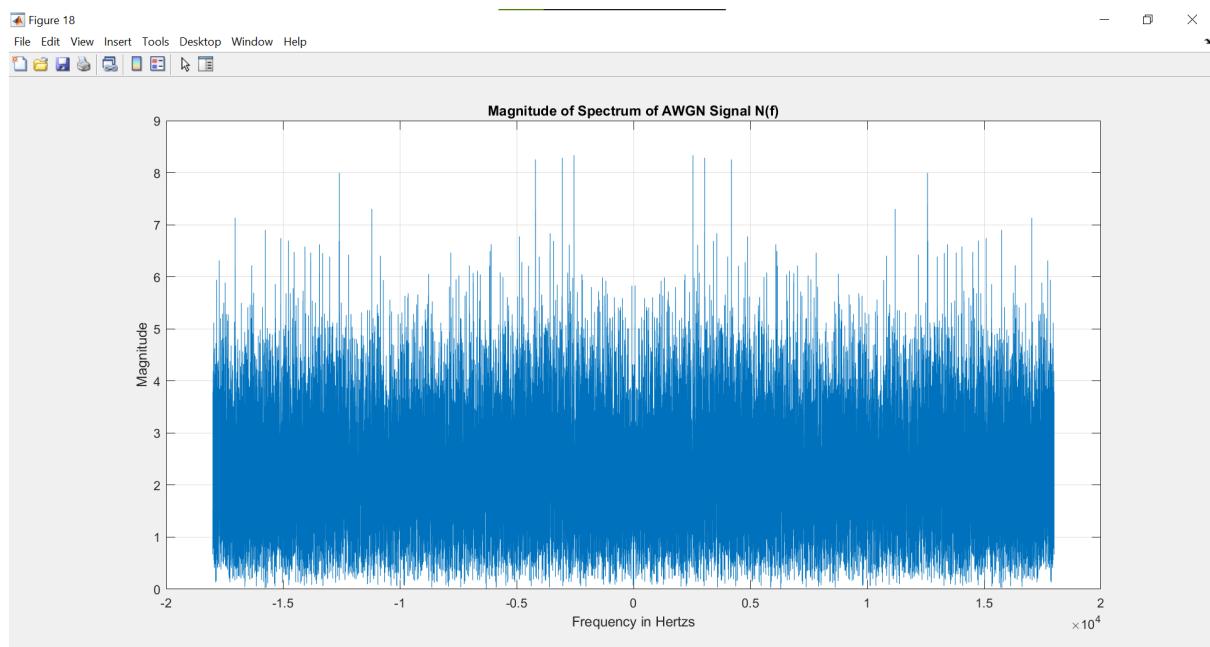
**Figure 59:** Magnitude of Spectrum of Modulated Signal  $X_c(f)$



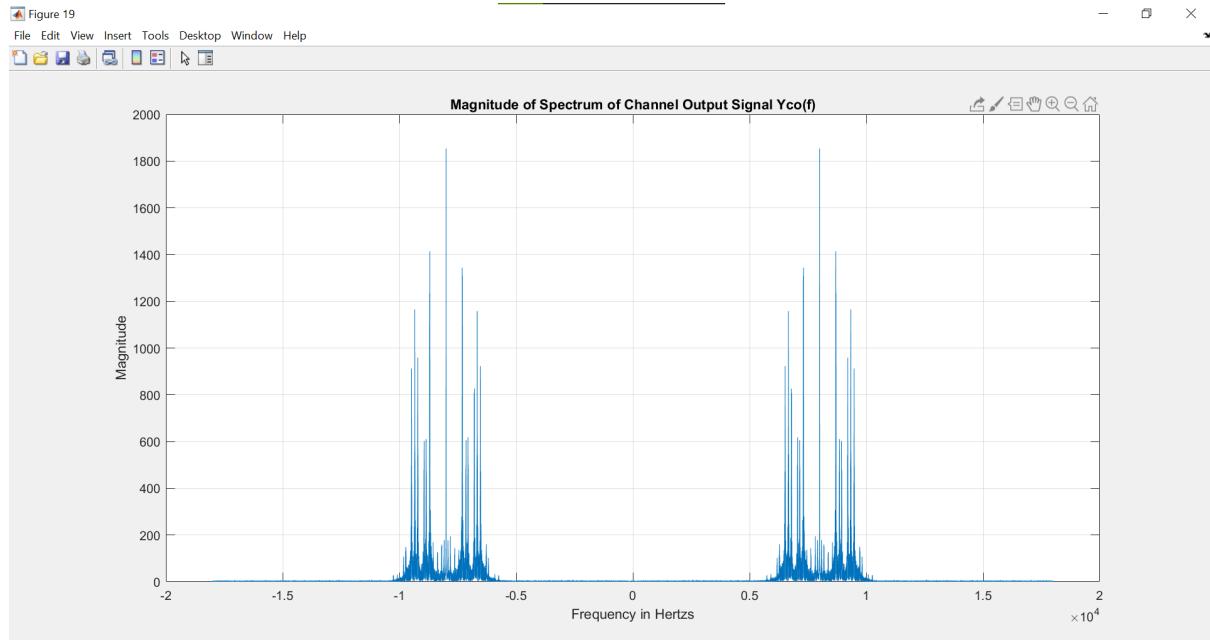
**Figure 60:** Magnitude of Spectrum of Channel Input Signal  $Y_{ci}(f)$



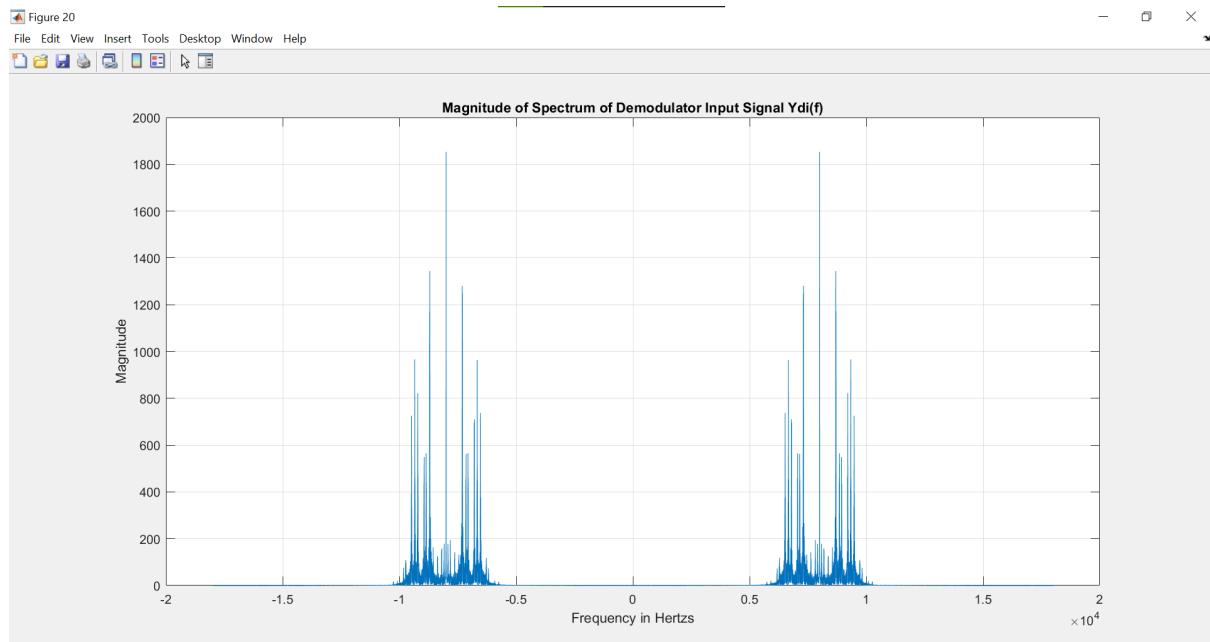
**Figure 61:** Magnitude of Spectrum of Channel Filter Output Signal  $Y_c(f)$



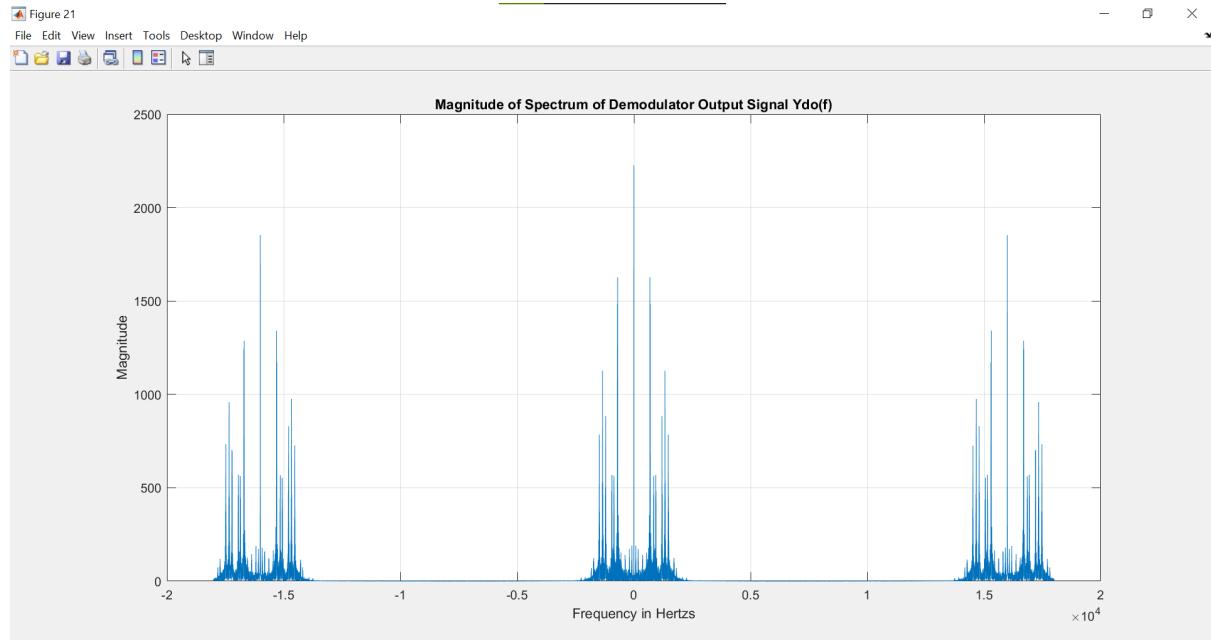
**Figure 62:** Magnitude of Spectrum of AWGN Signal  $N(f)$



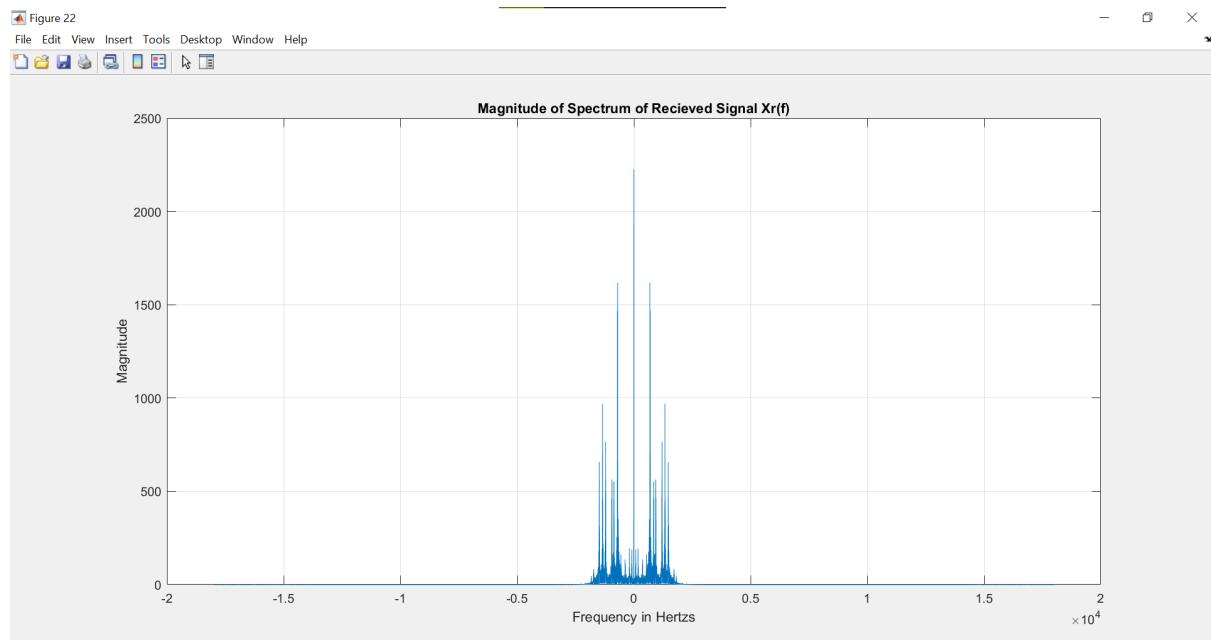
**Figure 63:** Magnitude of Spectrum of Channel Output Signal  $Y_{CO}(f)$



**Figure 64:** Magnitude of Spectrum of Demodulator Input Signal  $Y_{DI}(f)$

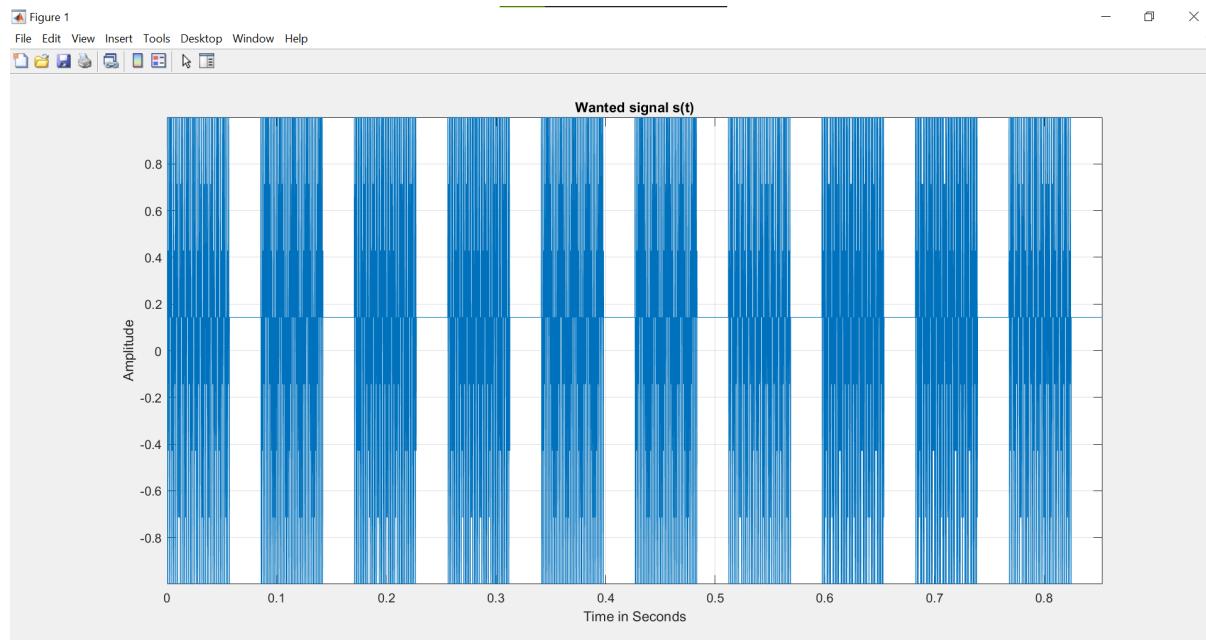


**Figure 65:** Magnitude of Spectrum of Demodulator Output Signal  $Y_{do}(f)$

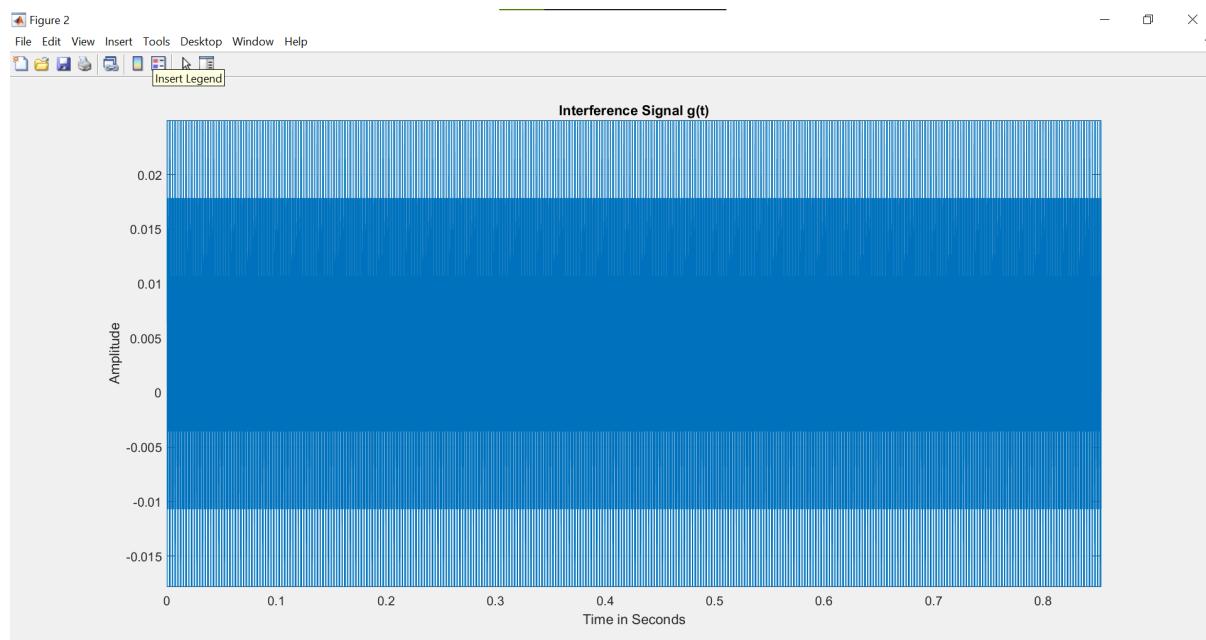


**Figure 66:** Magnitude of Spectrum of Received Signal  $X_r(f)$

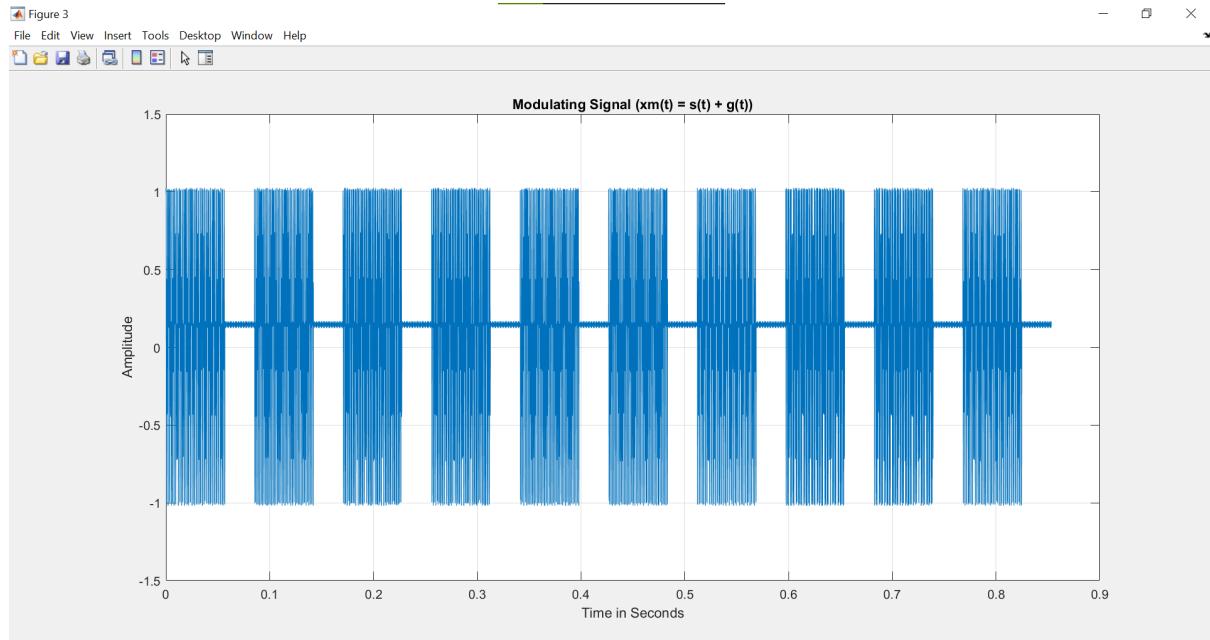
## 2.2



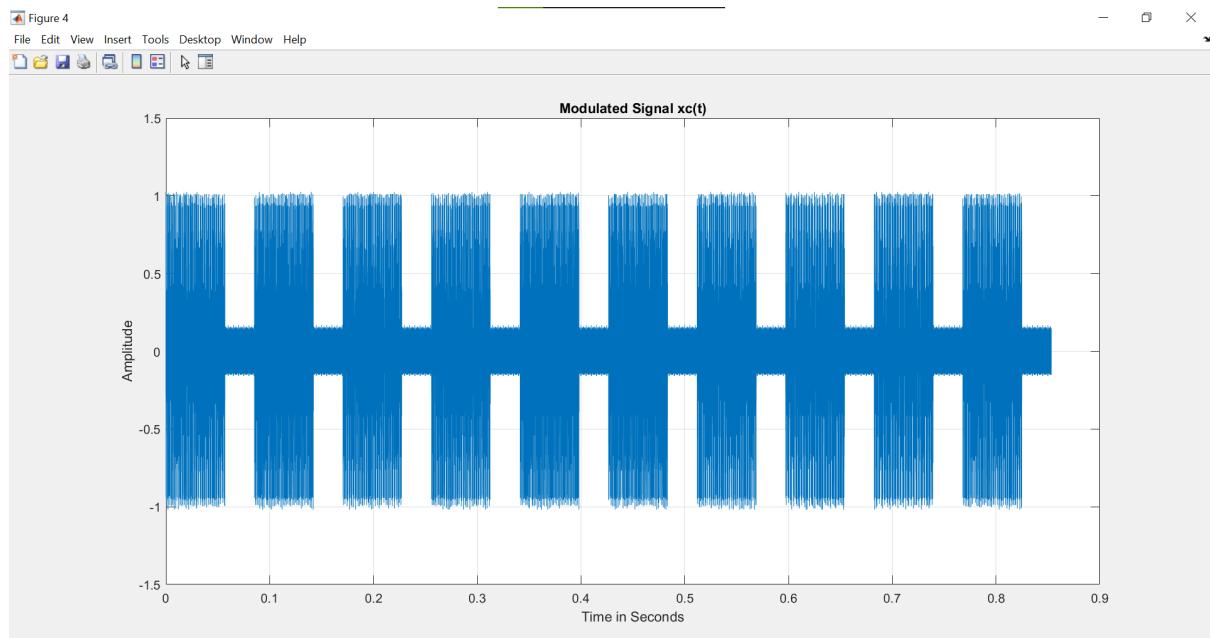
**Figure 67:** Wanted Signal  $s(t)$  in Time Domain



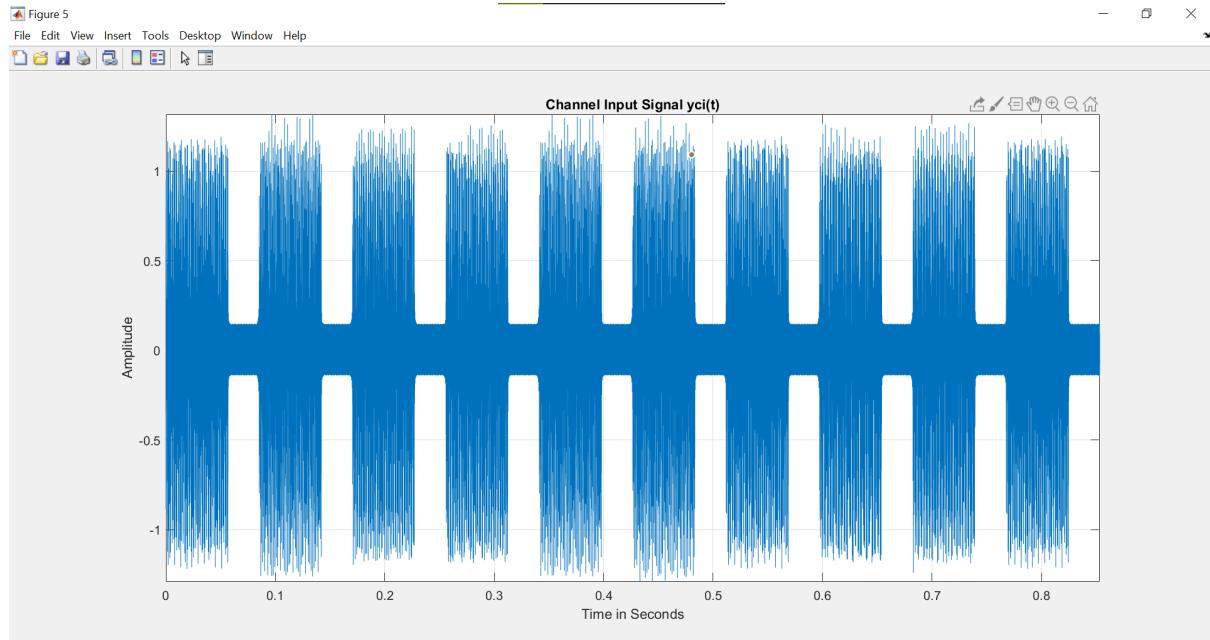
**Figure 68:** Interference Signal  $g(t)$  in the Time Domain



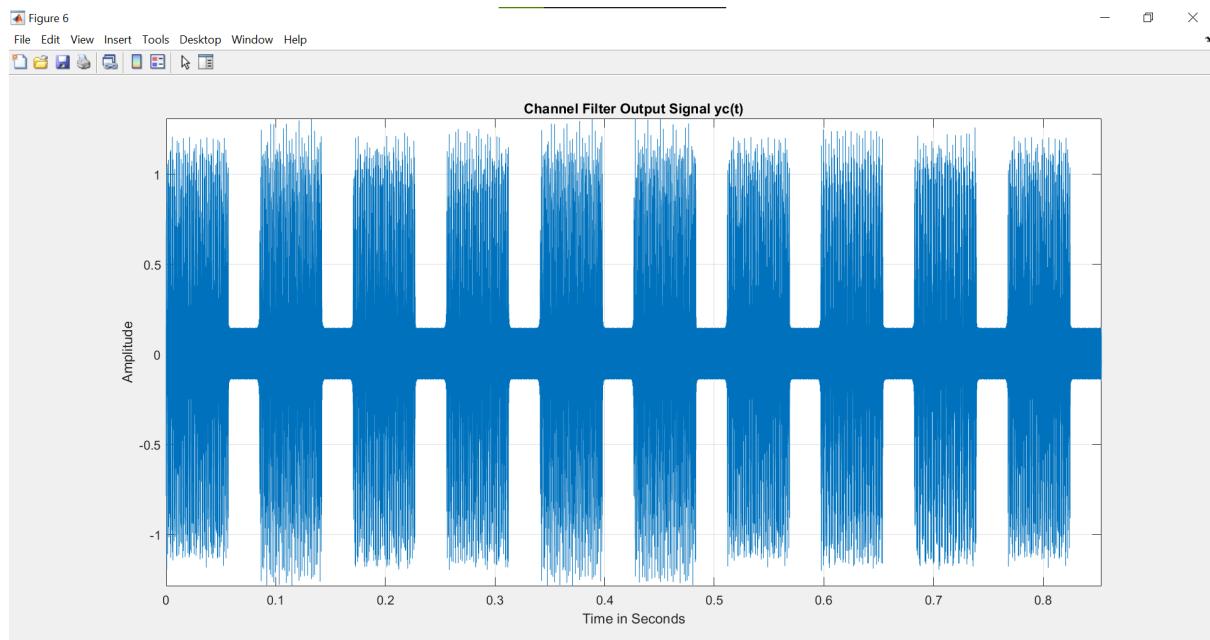
**Figure 69:** Modulating Signal ( $xm(t) = s(t) + g(t)$ )



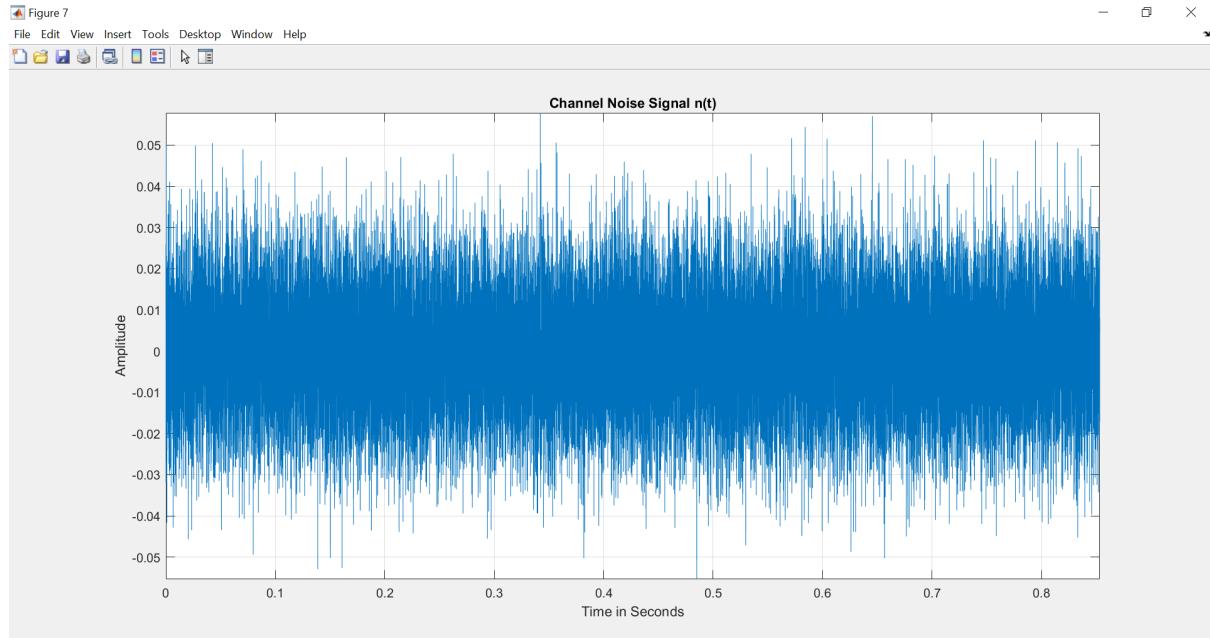
**Figure 70:** Modulated Signal  $xc(t)$  in the Time Domain



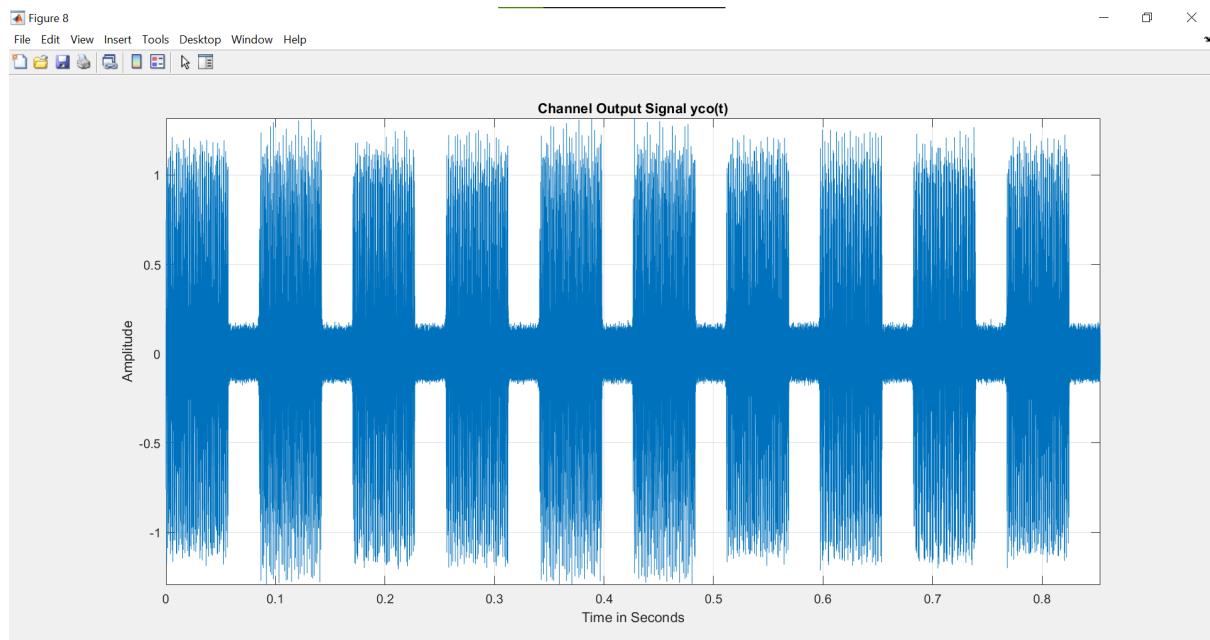
**Figure 71:** Channel Input Signal  $y_{ci}(t)$



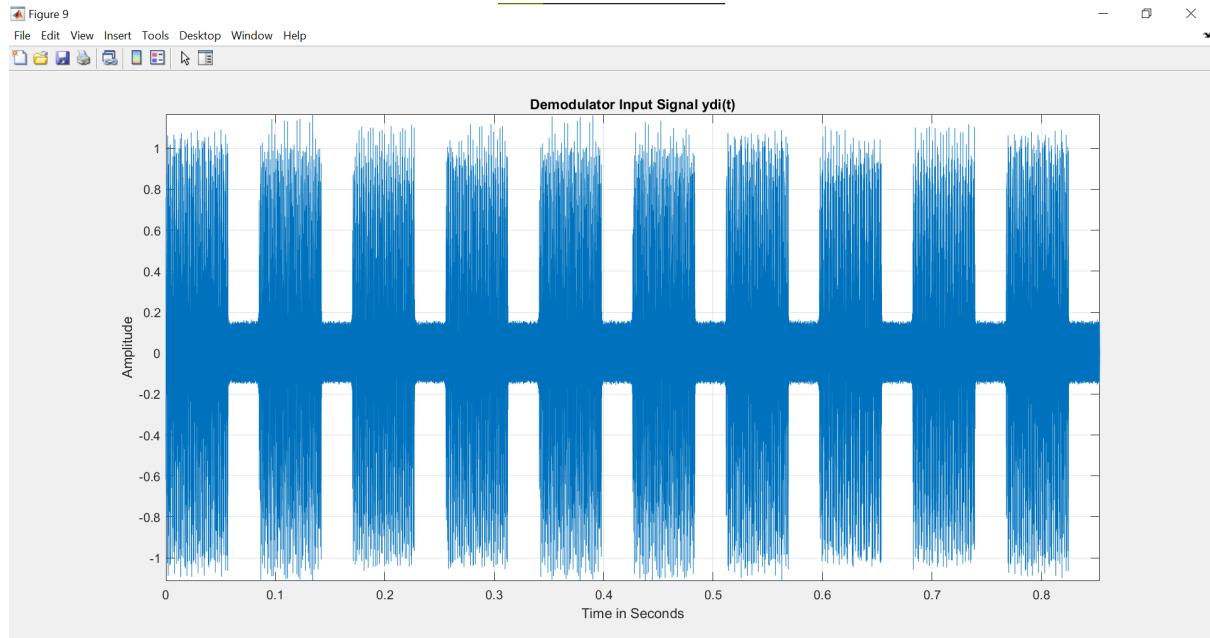
**Figure 72:** Channel Filter Output Signal  $y_c(t)$



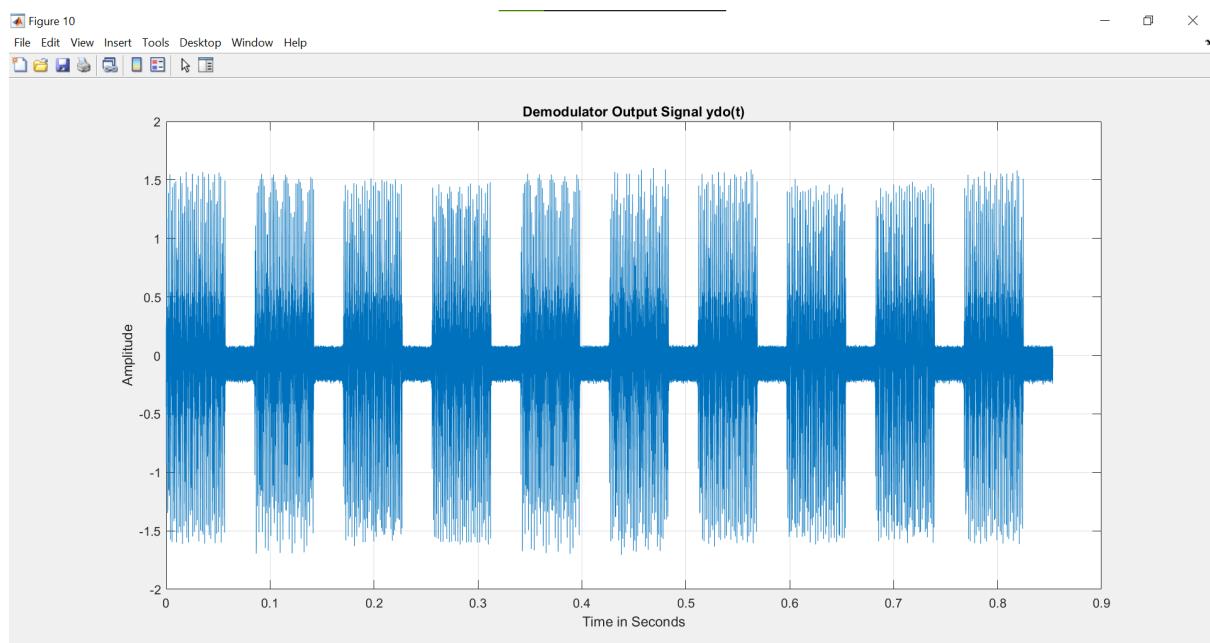
**Figure 73:** Channel Noise Signal  $n(t)$



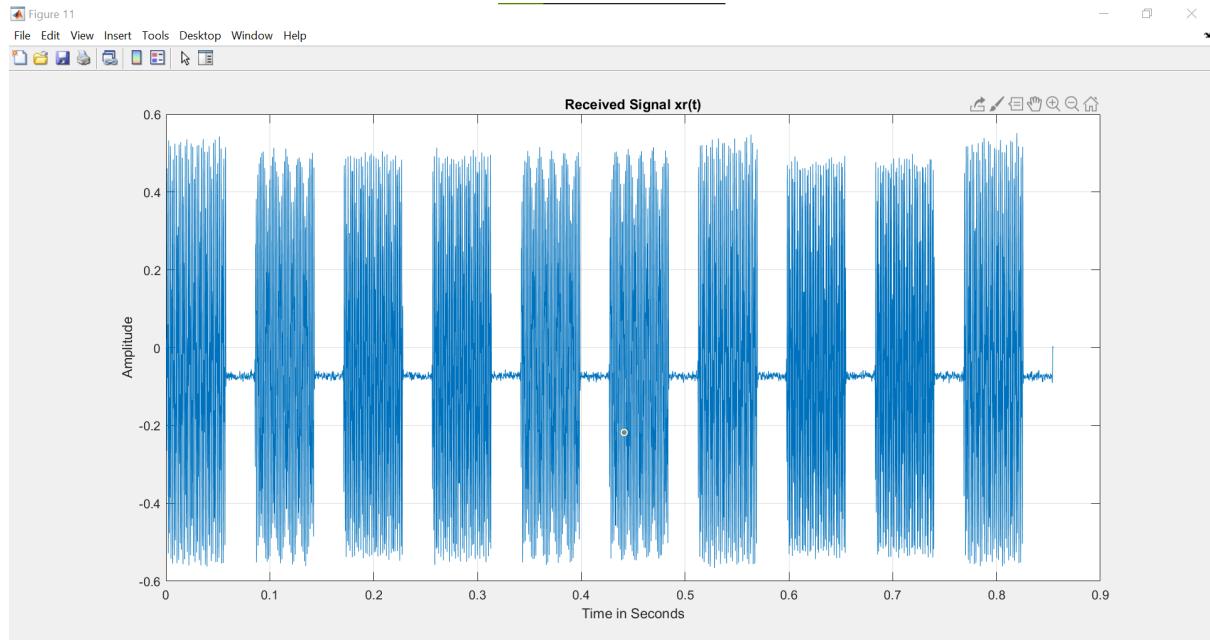
**Figure 74:** Channel Output Signal  $y_{co}(t)$



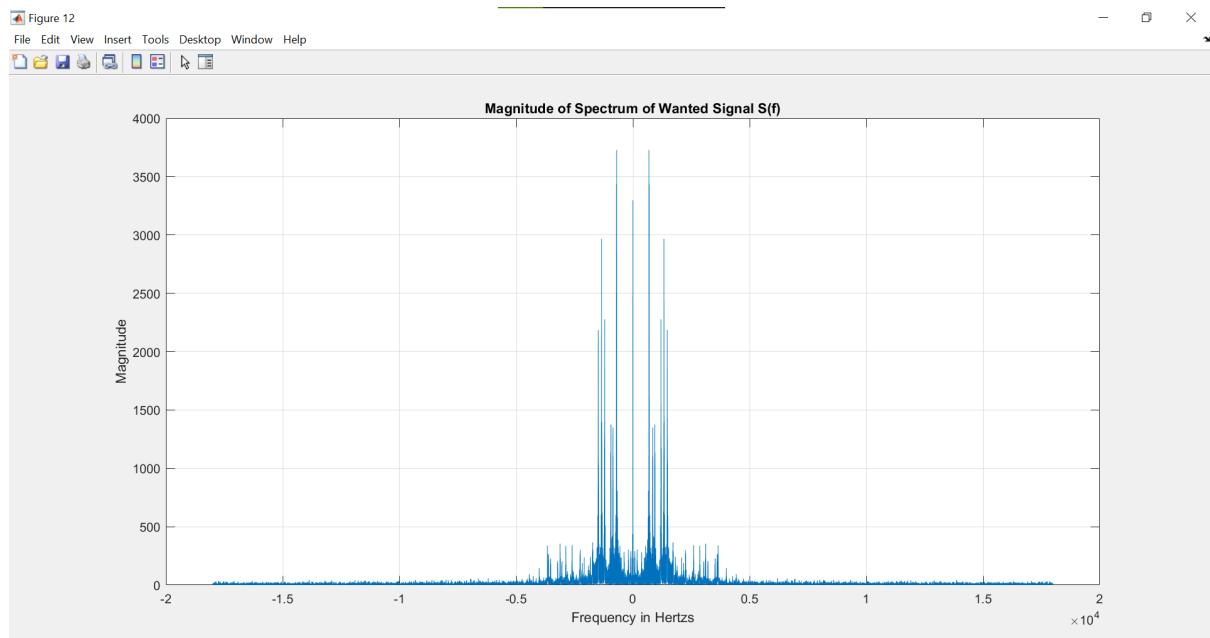
**Figure 75:** Demodulator Input Signal  $y_{di}(t)$



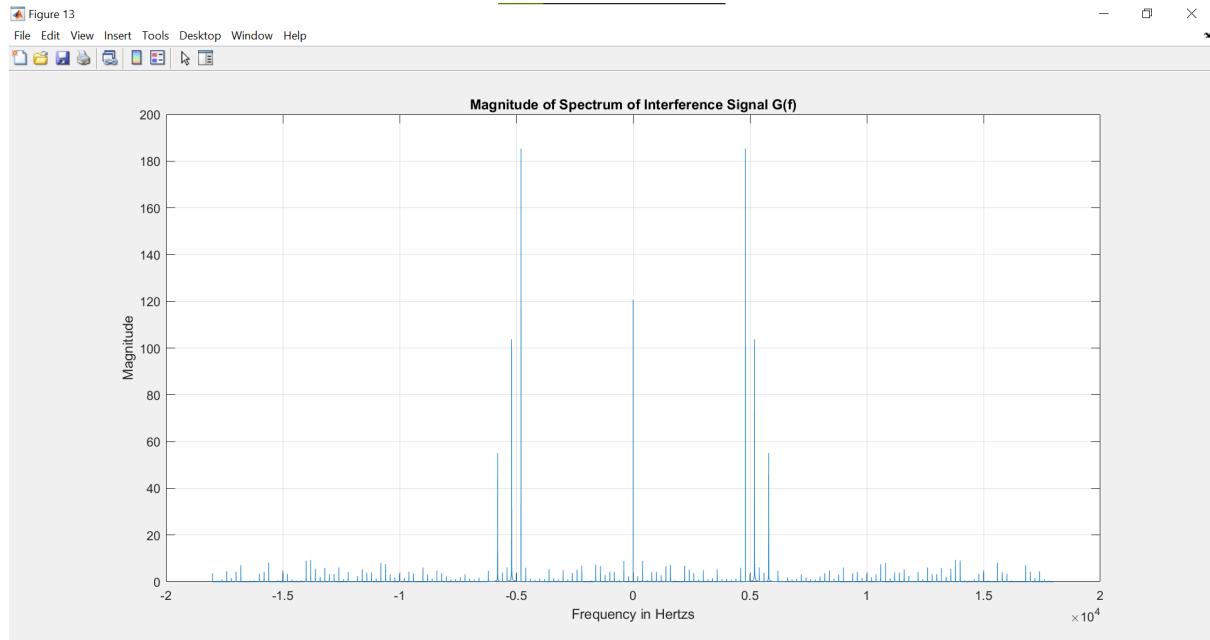
**Figure 76:** Demodulator Output Signal  $y_{do}(t)$



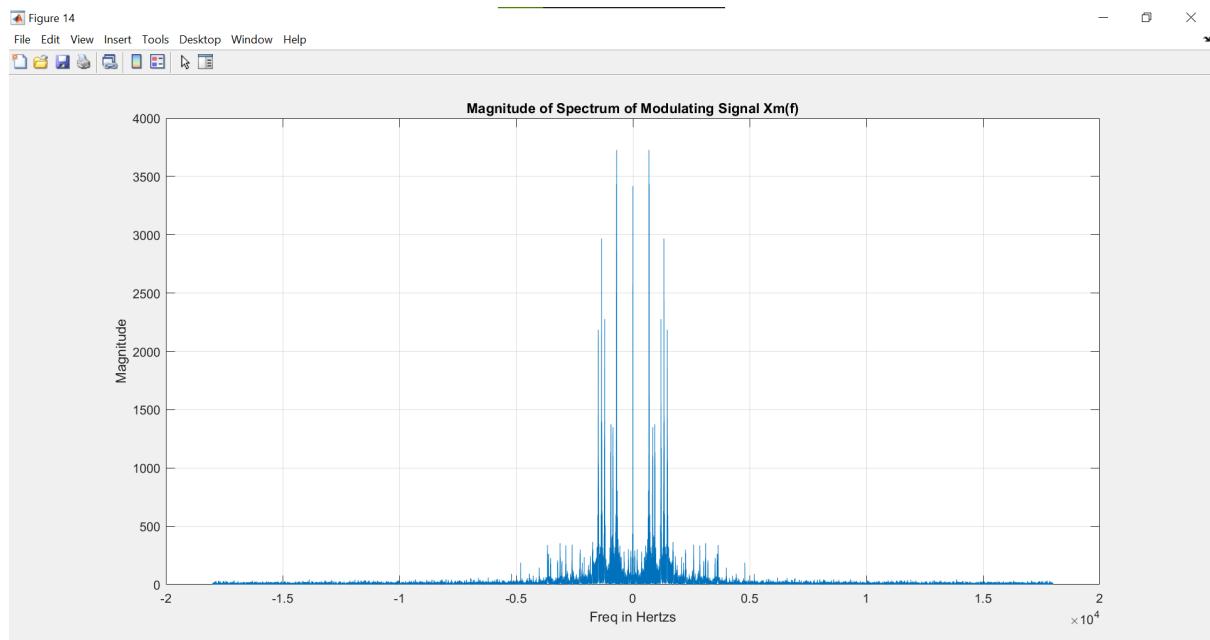
**Figure 77:** Received Signal  $x_r(t)$



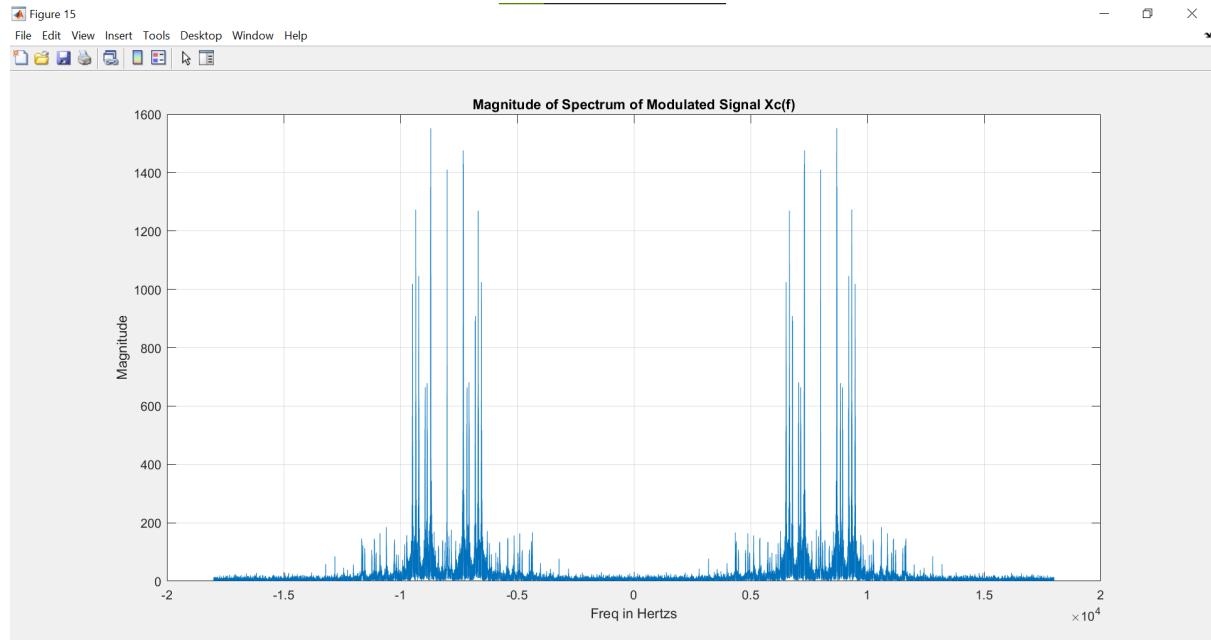
**Figure 78:** Magnitude of Spectrum of Wanted Signal  $S(f)$



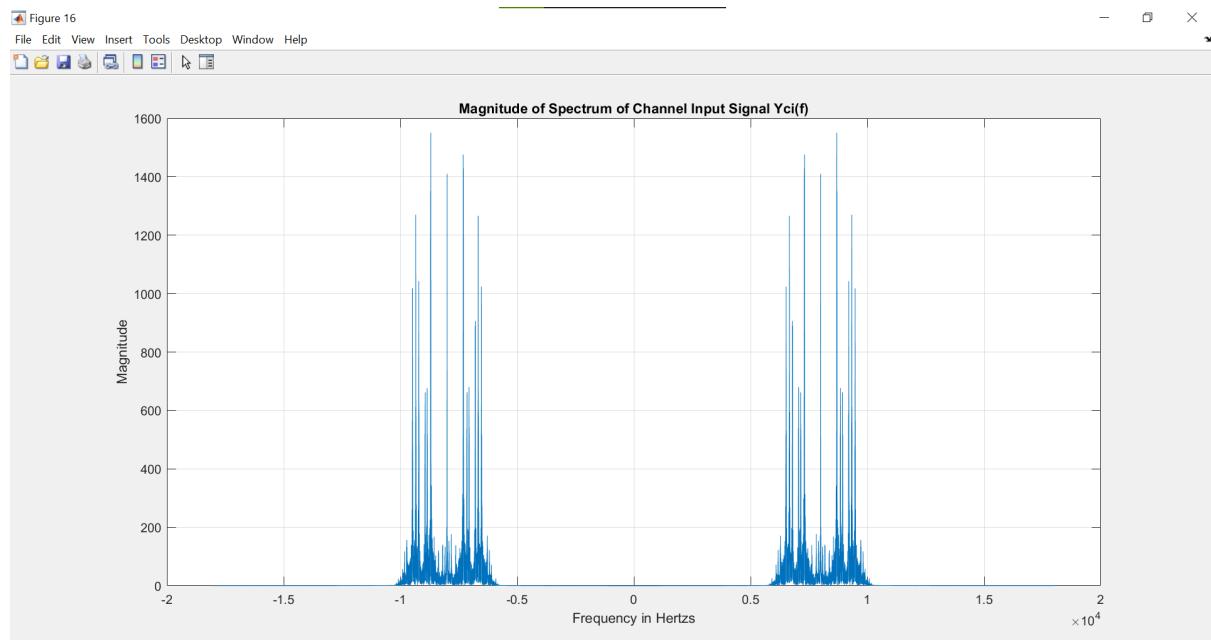
**Figure 79:** Magnitude of Spectrum of Interference Signal  $G(f)$



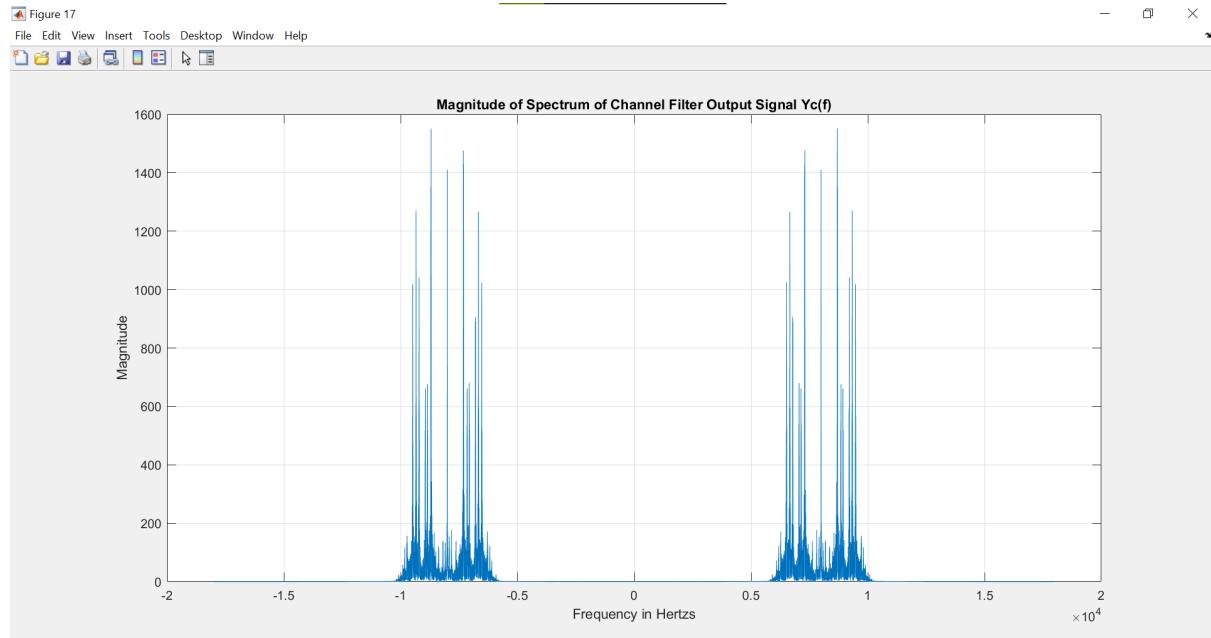
**Figure 80:** Magnitude of Spectrum of Modulating Signal  $Xm(f)$



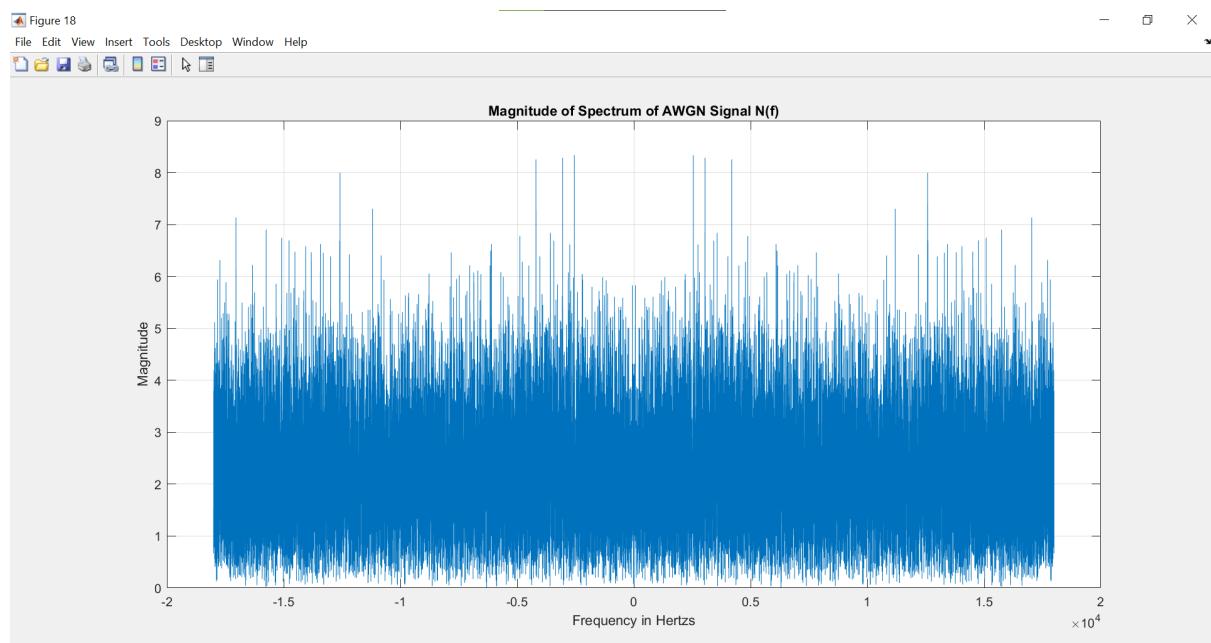
**Figure 81:** Magnitude of Spectrum of Modulated Signal  $X_c(f)$



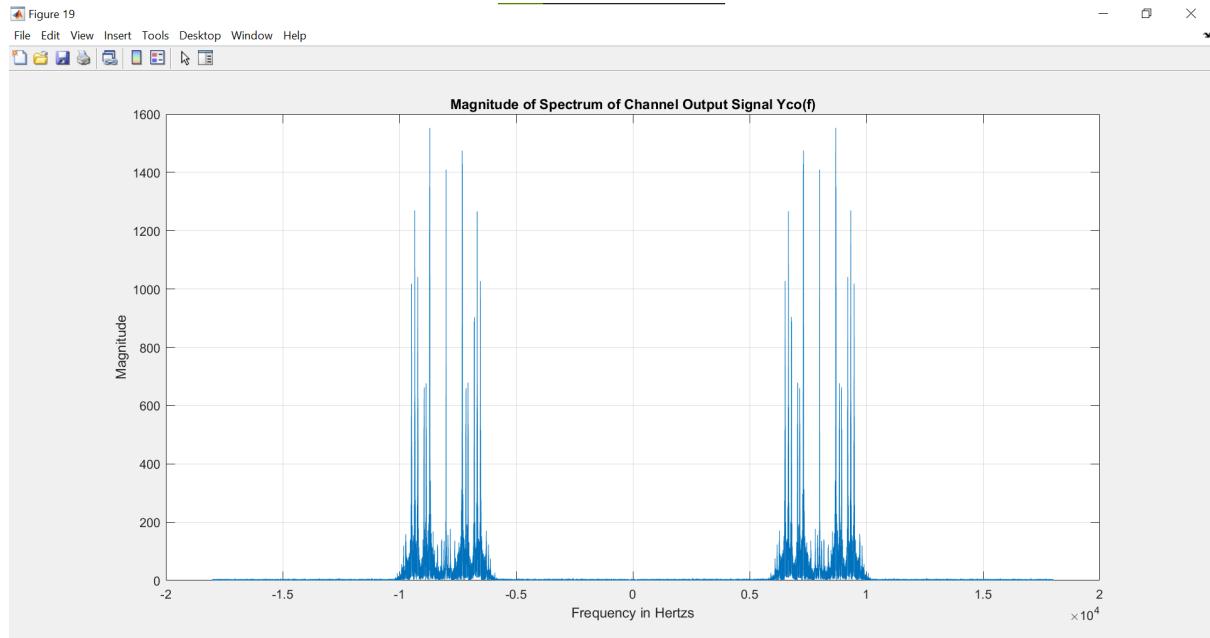
**Figure 82:** Magnitude of Spectrum of Channel Input Signal  $Y_{ci}(f)$



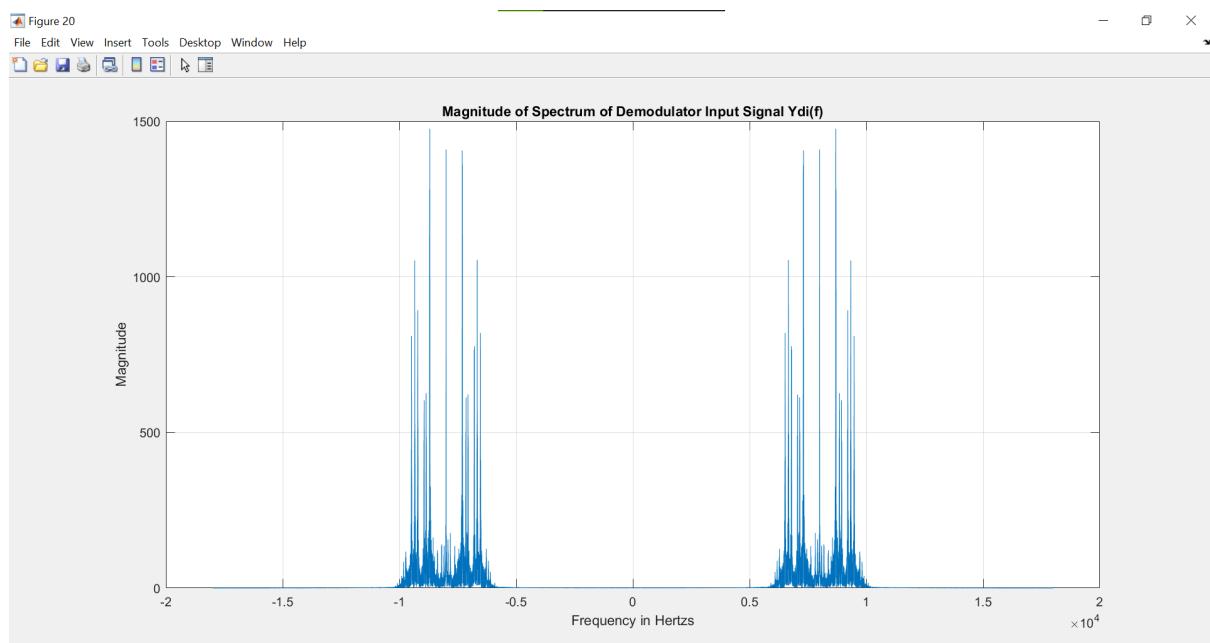
**Figure 83:** Magnitude of Spectrum of Channel Filter Output Signal  $Y_c(f)$



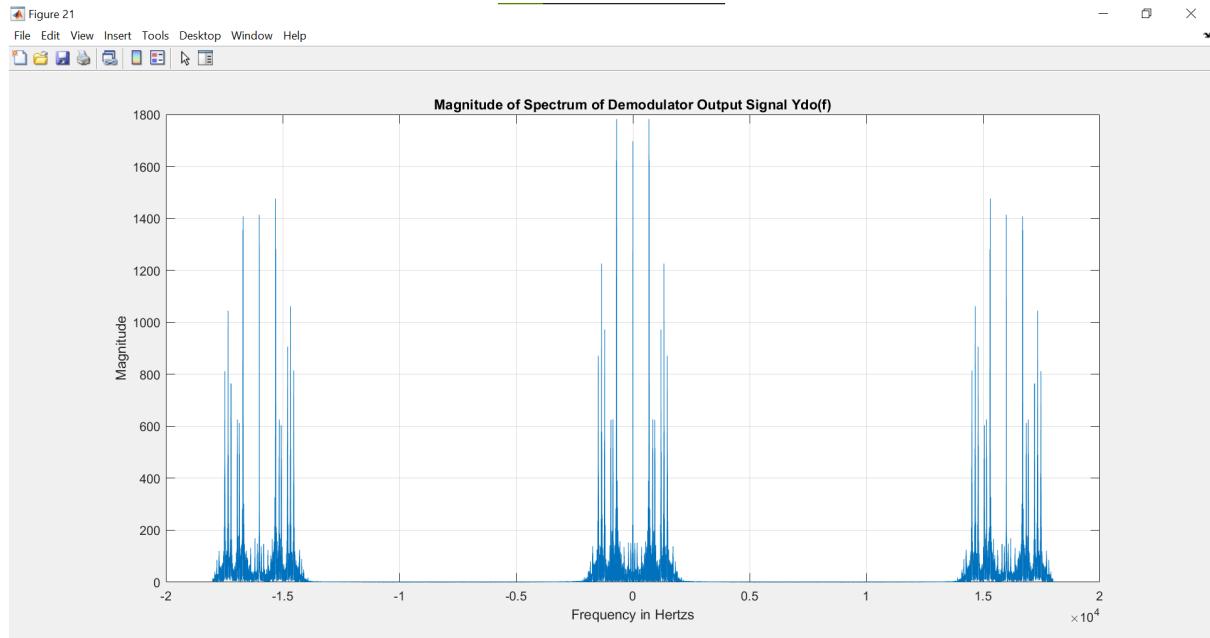
**Figure 84:** Magnitude of Spectrum of AWGN Signal  $N(f)$



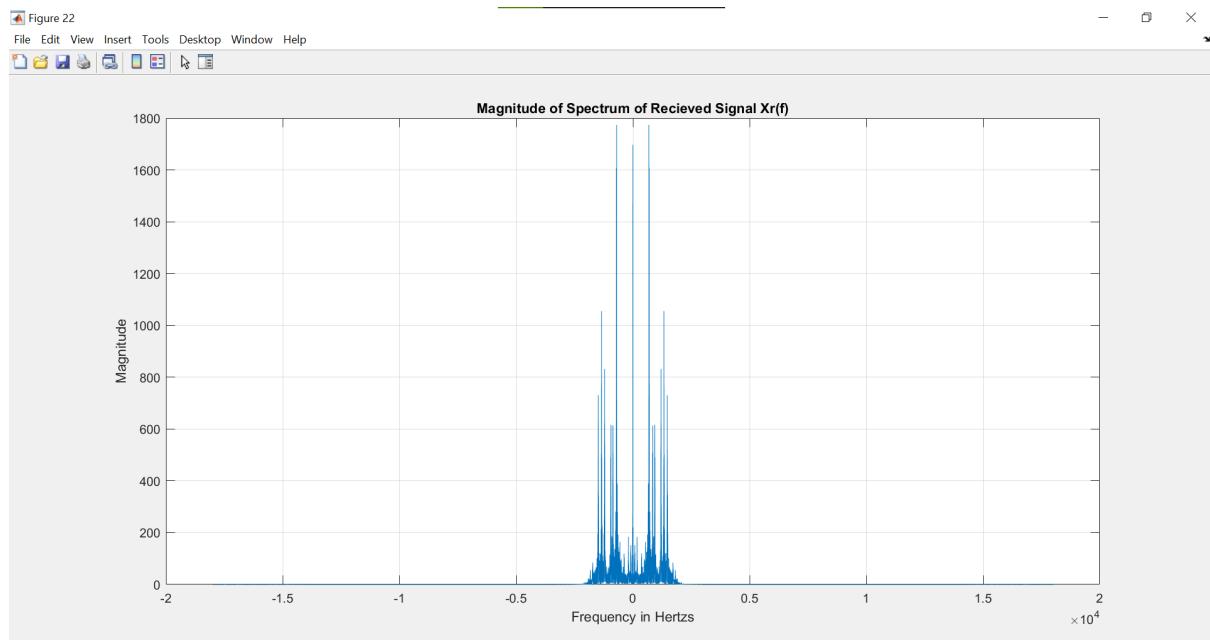
**Figure 85:** Magnitude of Spectrum of Channel Output Signal  $Y_{co}(f)$



**Figure 86:** Magnitude of Spectrum of Demodulator Input Signal  $Y_{di}(f)$



**Figure 87: Magnitude of Spectrum of Demodulator Output Signal  $Y_{do}(f)$**



**Figure 88: Magnitude of Spectrum of Received Signal  $X_r(f)$**

