אותות ומערכות-תרגיל מטלב 2\ אליהו אטין-205868771

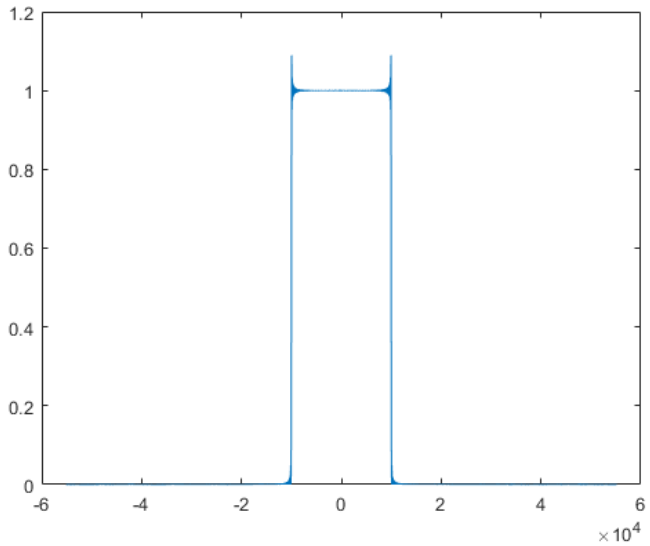
מטרת התרגיל: המערכת מדמה מערכת תקשורת בזמן רציף.

מערכת התקשורת מבוססת על שיטת האפנון DSB.

במערכת התקשורת ישנם שני אותות מידע אשר אנו נדרשים לשלב ולשדר דרך ערוץ שידור אחד ללא דריסות ועיוותים. לאחר קבלת האות המשודר נצטרך להוריד את שני האותות המשודרים לפס בסיס באמצעות מימוש מערכת קליטה.

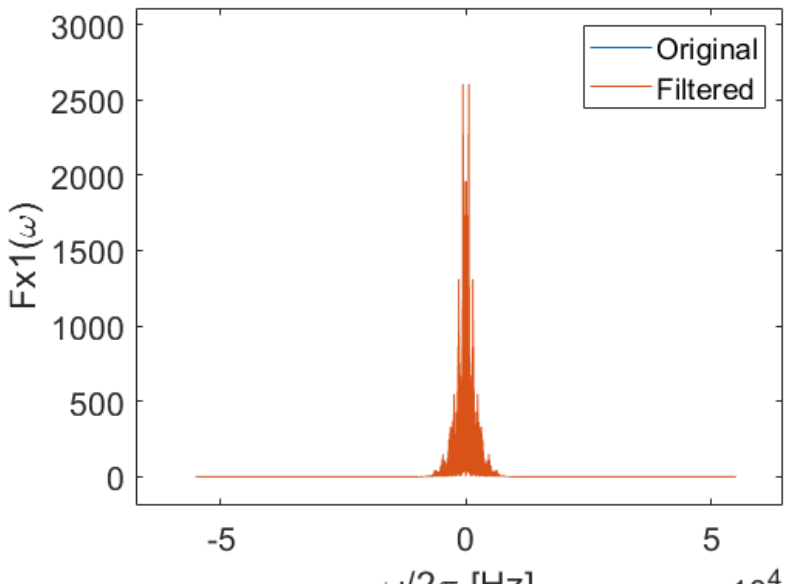
יצירת הפילטר

המסנן יהיה LPF עם תדר קטעון fc0=1.5*10^4[HZ] לשם קביעת fc0 הסתכלתי על אותות הכניסה ורוחב הסרט הכולל הזמין לערוץ השידור עבור 2 האותות.
התמרת הפורייה שלו תראה כחלון בגובה 1 בתדרים [-1,1].

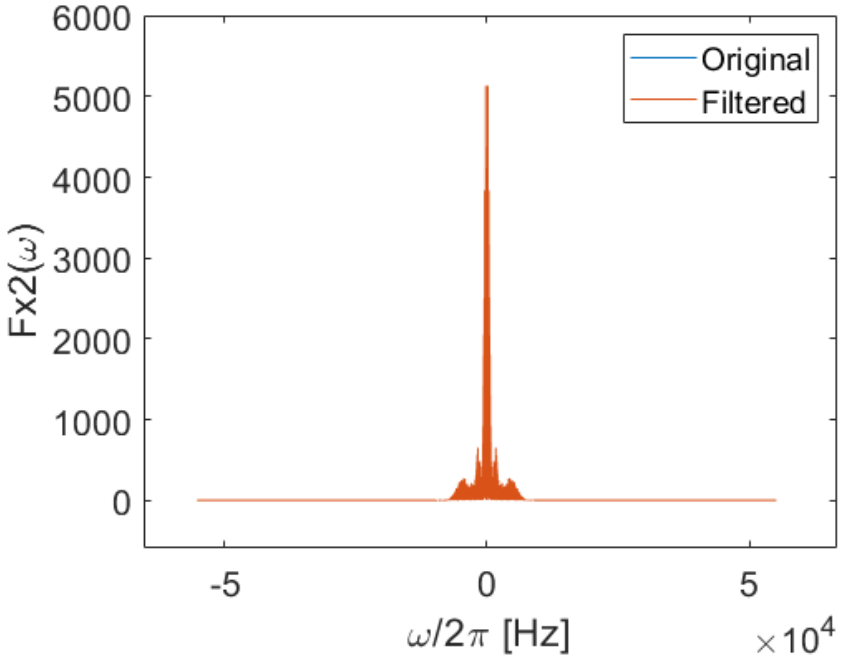


אותות הכניסה-לאחר מעבר בפילטר המסנן.

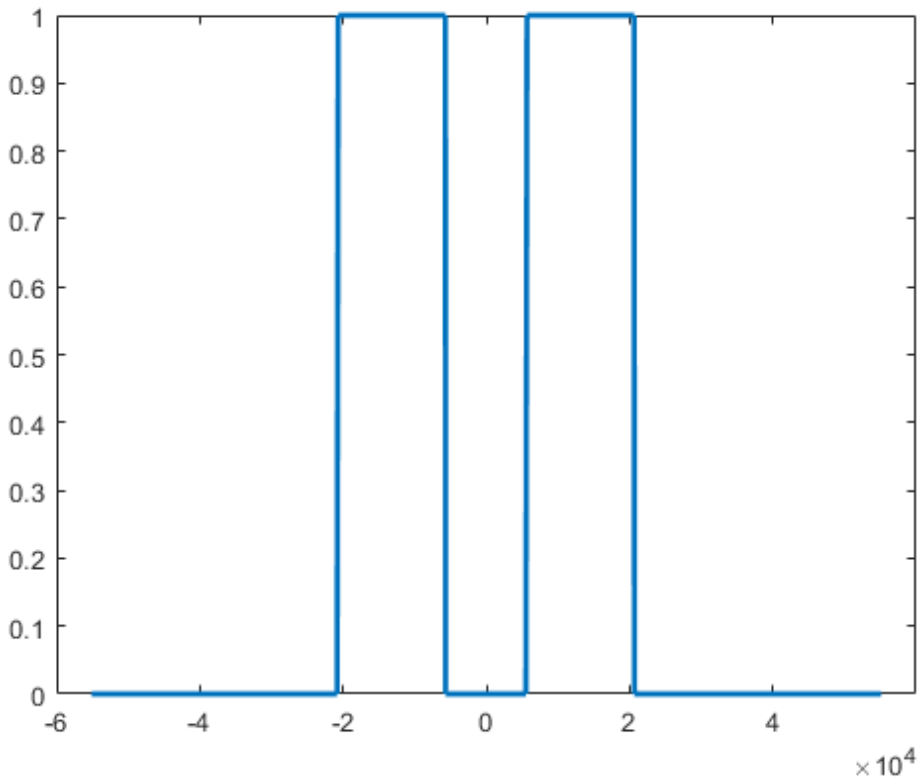
אות ראשון:



אות שני:



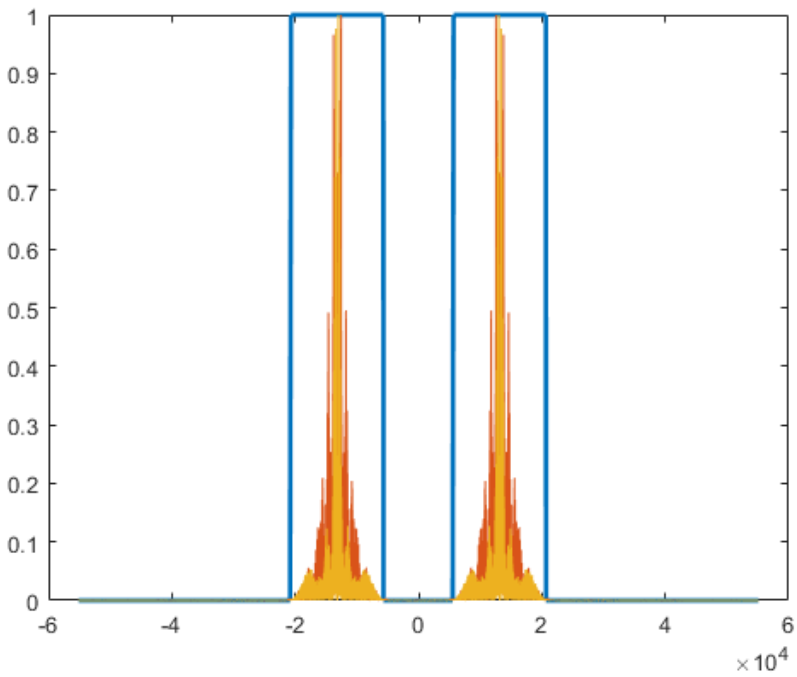
יצירת ערוץ התקשורת לשם שידור המידע: הערוץ שהוגרל לי לפי ת.ז שלי ID=205868771:



<u>שידור האותות בערוץ התקשורת:</u>

קבעתי את ערכי תדרי הגלים הנושאים fc1 fc2 להיות: fc1=fc2=1.3179105*10^4[HZ]

שידור האותות דרך הערוץ:



נחבר את שני האותות y=y1+y2, ונעביר אותם דרך הערוץ.

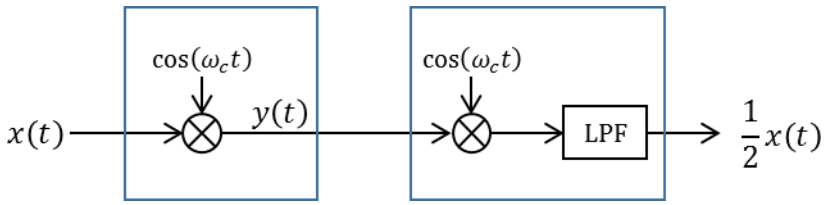
לאחר מכן נבצע דמודלציה לאותות שהתקבלו בצד של הreciever:

```
yr1 = y1.*cos(fc1*2*pi.*t);
yr2 = y2.*cos(fc1*2*pi.*t);
```

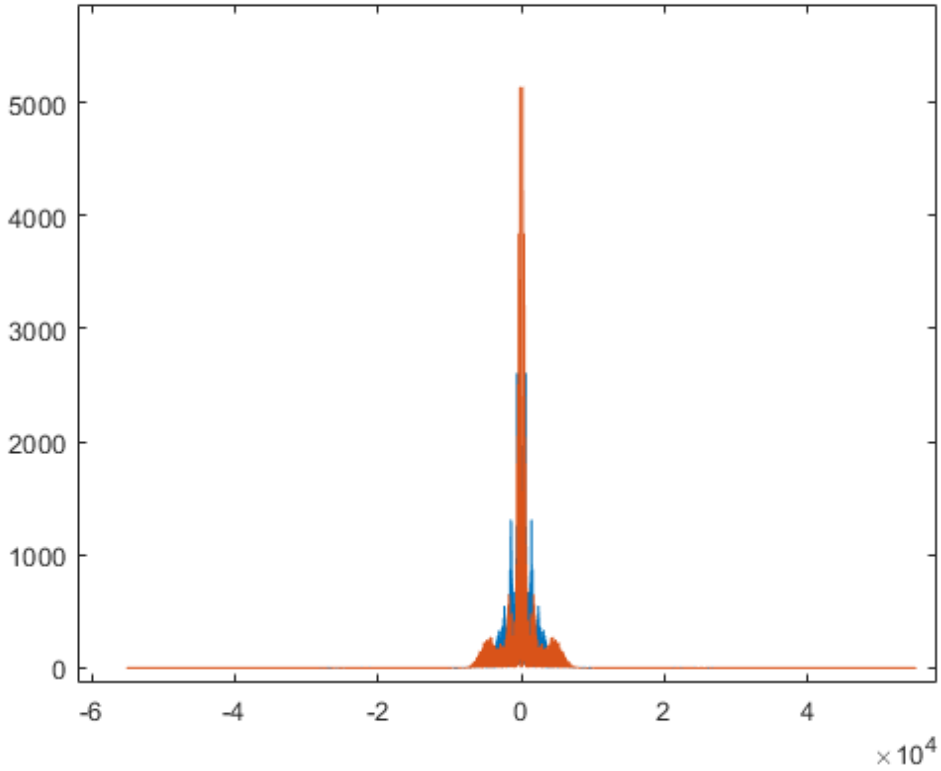
ונעביר את האותות שהתקבלו ב LPF ע"י פונקציית conv():

```
xr1 = conv(yr1,h_LPF,'same')*2;
xr2 = conv(yr2,h_LPF,'same')*2;
```

נכפיל ב2 כי נקבל חצי מהאות ביציאה לפי המודל:



שנכון גם אם עבור שני אותות כניסה שכל אחד מוכפל בגל הנושא ומחברים את שניהם לפני ההעברה בערוץ השידור בצד המקבל לאחר ההכפלה של האותות שהתקבלו ב cos(fc1*2*pi*t) ומעבר דרך המסנן יש להכפיל ב2 כדי לקבל את חיבור שני האותות המועברים כדרוש.
נעשה לאותות התמרת פורייה ונראה את האות שקיבלנו בצד המקבל(אמור להיות חיבור של שני האותות x1 וx2 שבכניסה).



קוד המטלה המלא:

```
close all
clc
clear
%% Generation of input signal
[t, x1,x2, Fs] = inputBuilder();
% soundsc(x,Fs)
%% Fourier transform of input signal
Nt = length(t);
fmax = Fs/2;
fvec = fmax*linspace(-1,1,Nt+1);
fvec = fvec(1:end-1);
wvec = 2*pi*fvec;
Fx1 = fftshift(fft(x1));%up to a factor of dt.
Fx2 = fftshift(fft(x2));
%%
figure(1);plot(fvec,abs(Fx1)');
xlabel('\omega/{2\pi} [Hz]','fontsize',16);
ylabel('Fx1(\omega)','fontsize',16);
set(gca,'fontsize',16);hold on;

figure(5);plot(fvec,abs(Fx2)');
xlabel('\omega/{2\pi} [Hz]','fontsize',16);
ylabel('Fx2(\omega)','fontsize',16);
set(gca,'fontsize',16);hold on;

%% Filter signal to the available BW of the channel
```

```
f_co =1*10^4;      %????;%Hz %%LPF cut-off frequency

omega_0 = 2*pi*f_co/Fs; %Scaling the filter to the discrete time
n=-1000:1:1000;
h_LPF = sin(omega_0*n)./(pi*n);
h_LPF(1001) = omega_0/pi;
H_LPF = fftshift(fft(h_LPF,Nt));
figure(7);plot(fvec,abs(H_LPF));
%% Pass the input signals through the LPF
x_f1 = conv(x1,h_LPF,'same');
x_f2 = conv(x2,h_LPF,'same');

Fx_f1 = fftshift(fft(x_f1));
figure(1);plot(fvec,abs(Fx_f1)');
legend('Original','Filtered')

Fx_f2 = fftshift(fft(x_f2));
figure(5);plot(fvec,abs(Fx_f2)');
legend('Original','Filtered')


%% Communication channel - Add your ID number
%%%%%%%
%In this part we generate the communication channel
% The channel supports transmission of data in a specific range of
% frequencies.
ID = 205868771;            %?????
Channel = channelBuilder(ID,fvec);
figure(2);plot (fvec,abs(Channel),'linewidth',2);
%%

fc1 = 1.3179105*10^4;                %????;%[Hz]
fc2 = 1.3179105*10^4;                 %????;%[Hz]

% Modulate the signals with AM modulation
y1 = x1.*cos(fc1*2*pi.*t);   % ????
y2 = x2.*cos(fc2*2*pi.*t);   %????;

Fy1 = fftshift(fft(y1));
Fy2 = fftshift(fft(y2));

figure(2);hold on;
plot(fvec,abs(Fy1)'/max(abs(Fy1)),fvec,abs(Fy2)'/max(abs(Fy2)));

y = y1+y2;
%% Passing the signal in the channel
y_r = simulateChannel(y,ID);
Y_r = fftshift(fft(y_r));
figure();plot(fvec,abs(Y_r));
%% Receiver
% de-modulate
yr1 = y1.*cos(fc1*2*pi.*t);    %????;
yr2 = y2.*cos(fc1*2*pi.*t);   %????;
% Filter
xr1 = conv(yr1,h_LPF,'same')*2;     %????;
xr2 = conv(yr2,h_LPF,'same')*2;   %????;

 %soundsc(xr1,Fs)
 %pause(2.5)
```

```
 %soundsc(xr2,Fs)
Fxr1 = fftshift(fft(xr1));
Fxr2 = fftshift(fft(xr2));
figure(3);plot(fvec,abs(Fxr1),fvec,abs(Fxr2)');
%% Check results
%Normalization
xr1n = xr1*sqrt(mean(x_f1.^2))/sqrt(mean(xr1.^2));
xr2n = xr2*sqrt(mean(x_f2.^2))/sqrt(mean(xr2.^2));
%%%%  MMSE

MMSE_1= 10*log10(mean(x_f1(:).^2)) - 10*log10(mean((x_f1(:)-
xr1n(:)).^2));
MMSE_2= 10*log10(mean(x_f2(:).^2)) - 10*log10(mean((x_f2(:)-
xr2n(:)).^2));

%% Performace evaluation:
[Grade]=GradeMyOutput(ID,y,f_co,fc1,fc2);
```