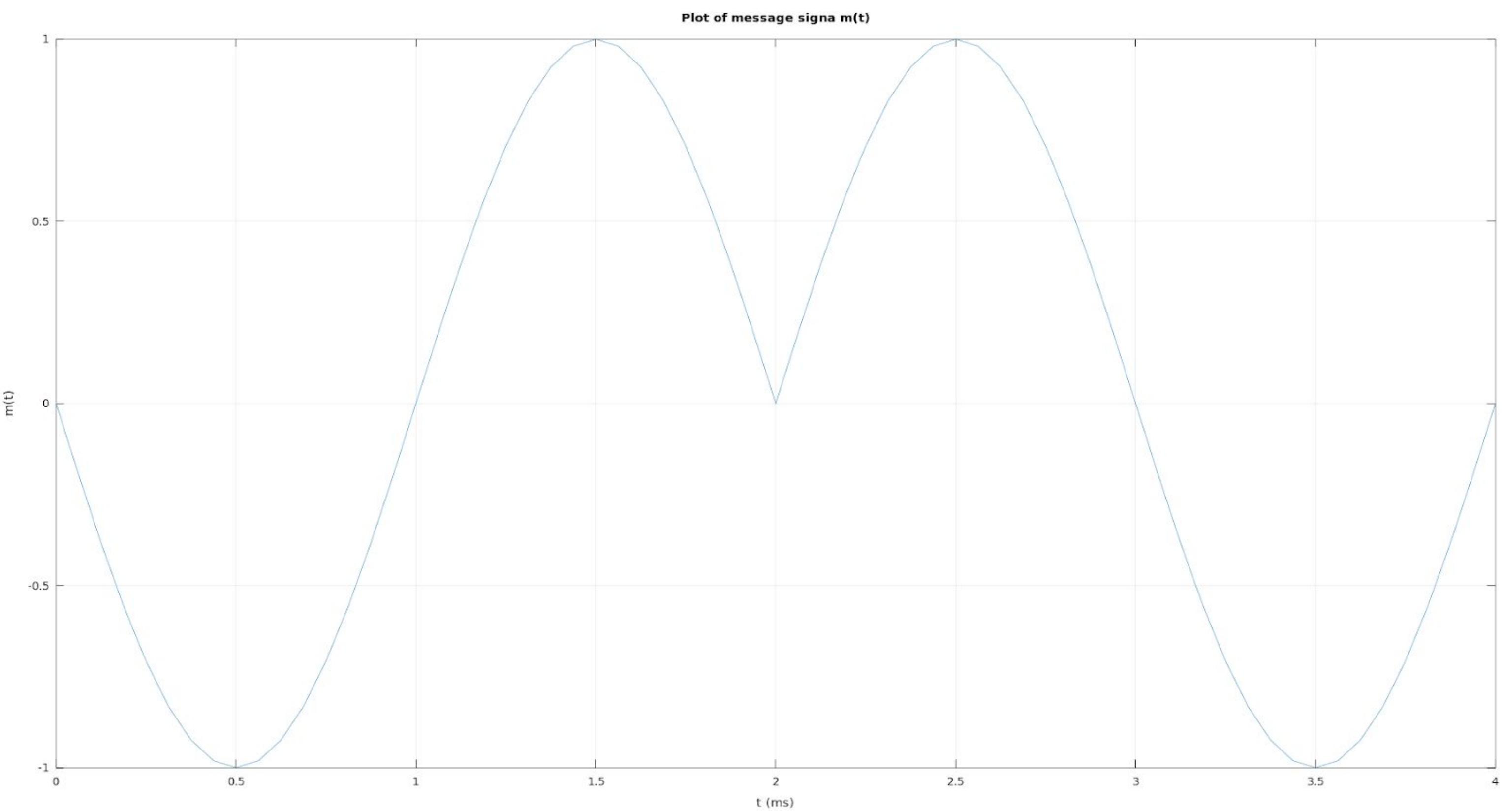
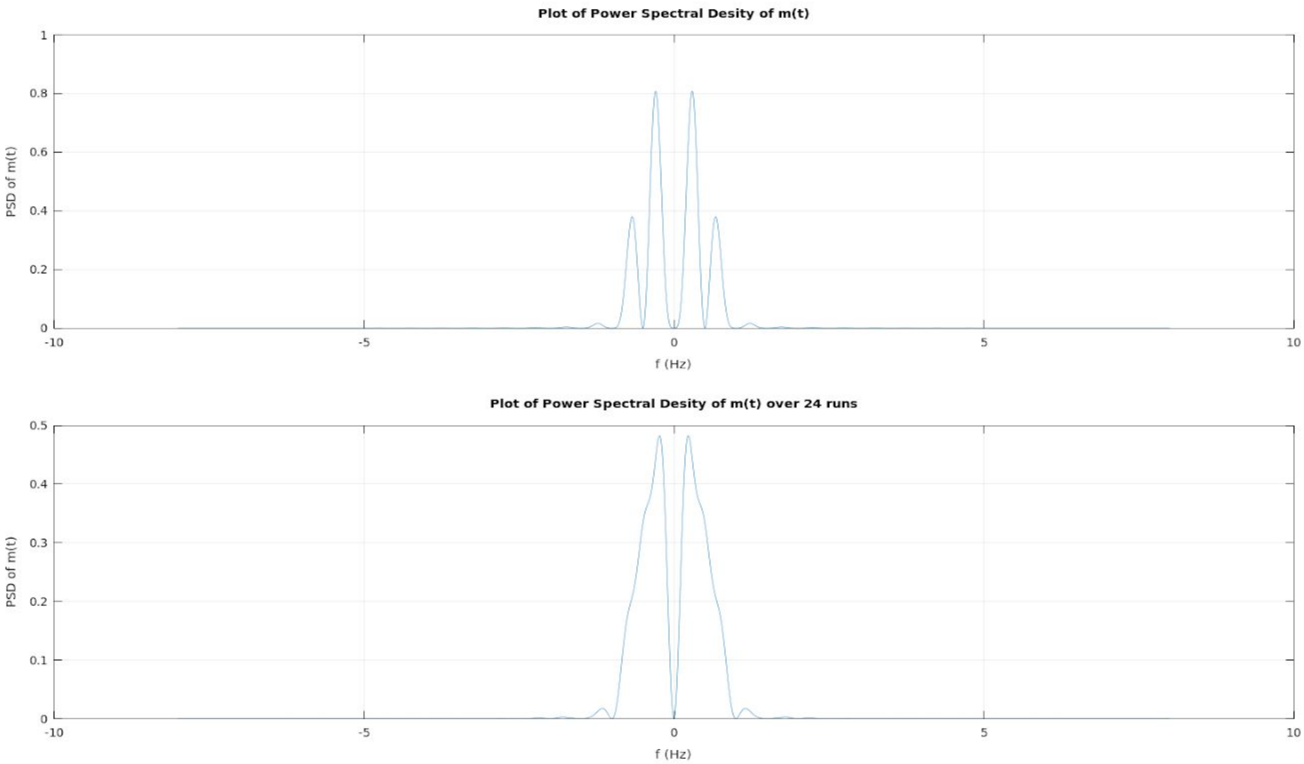
Lab Report-4

By: Ronak Doshi (IMT2017523)

Message Signal

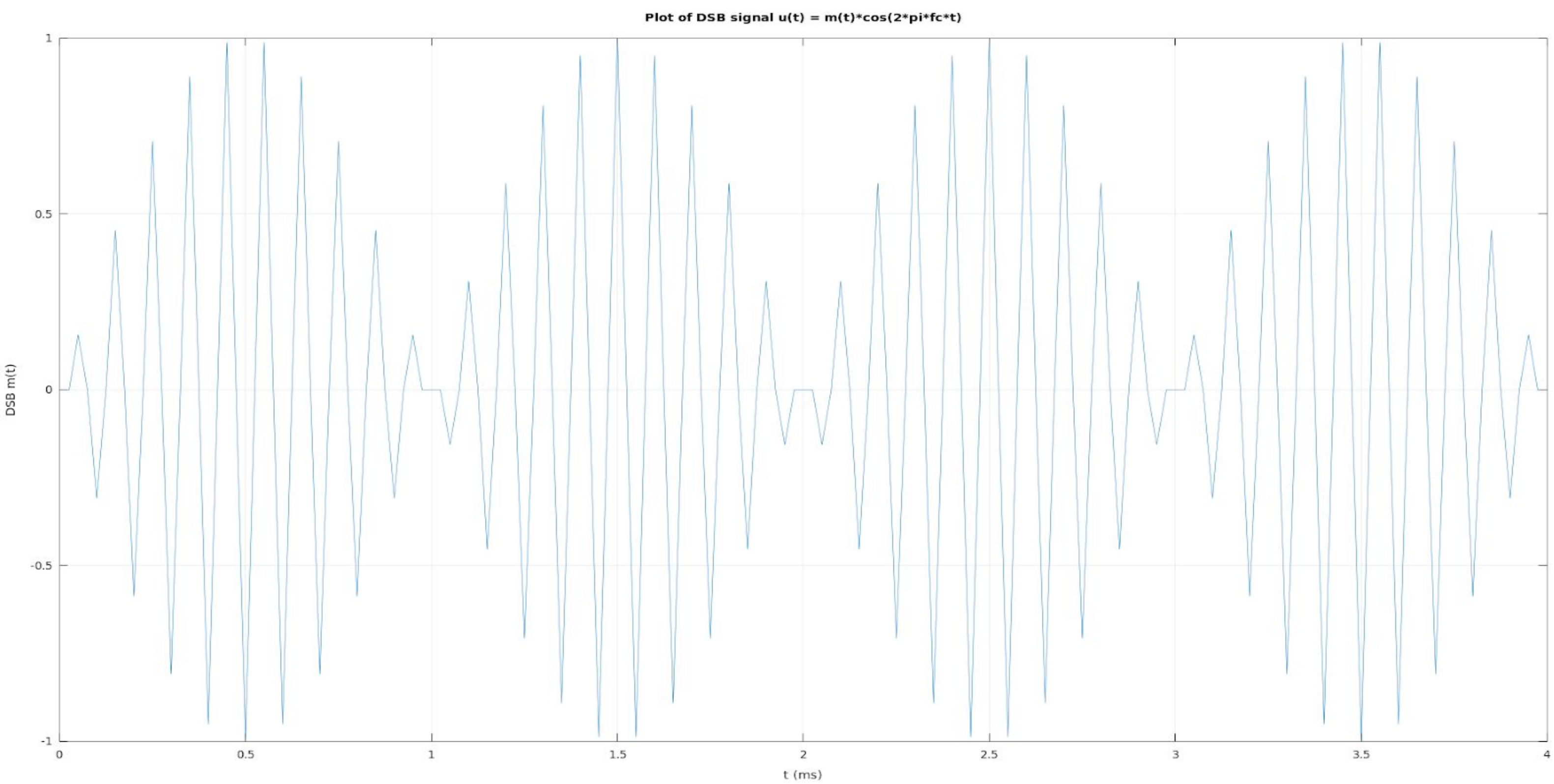


PSD of Message Signal

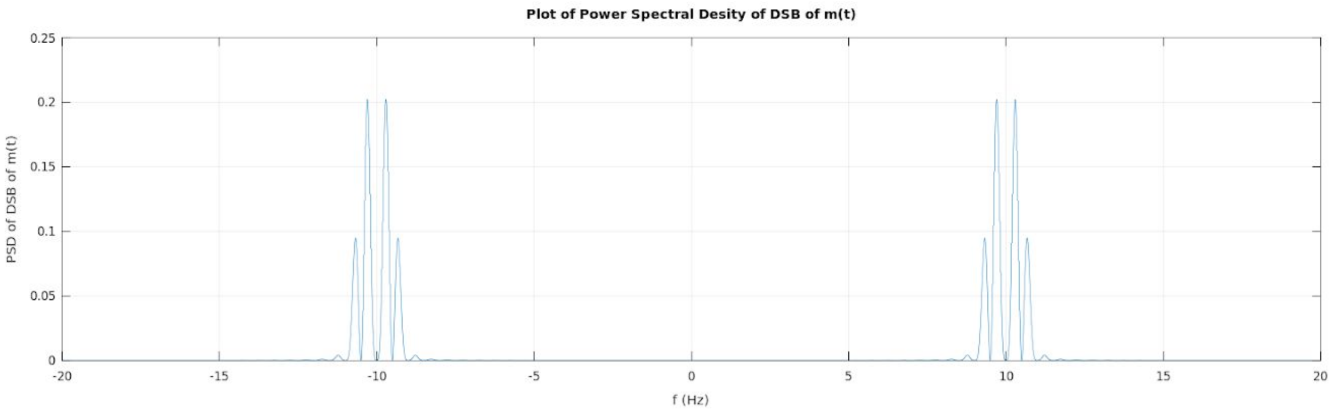


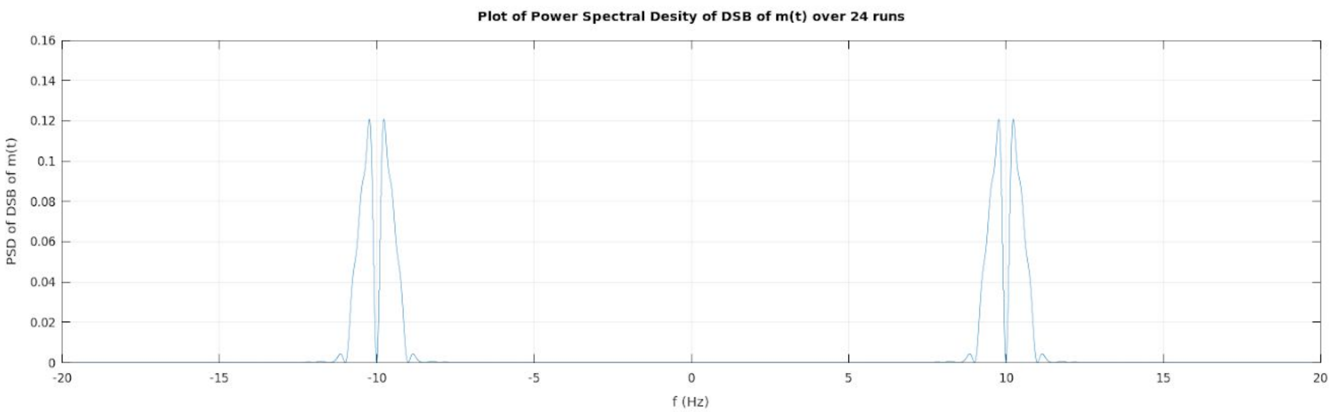
Yes, You can Eyeball the bandwidth of the signal. Bandwidth comes out to be 2 as values beyond that are zero.

DSB of the Message Signal



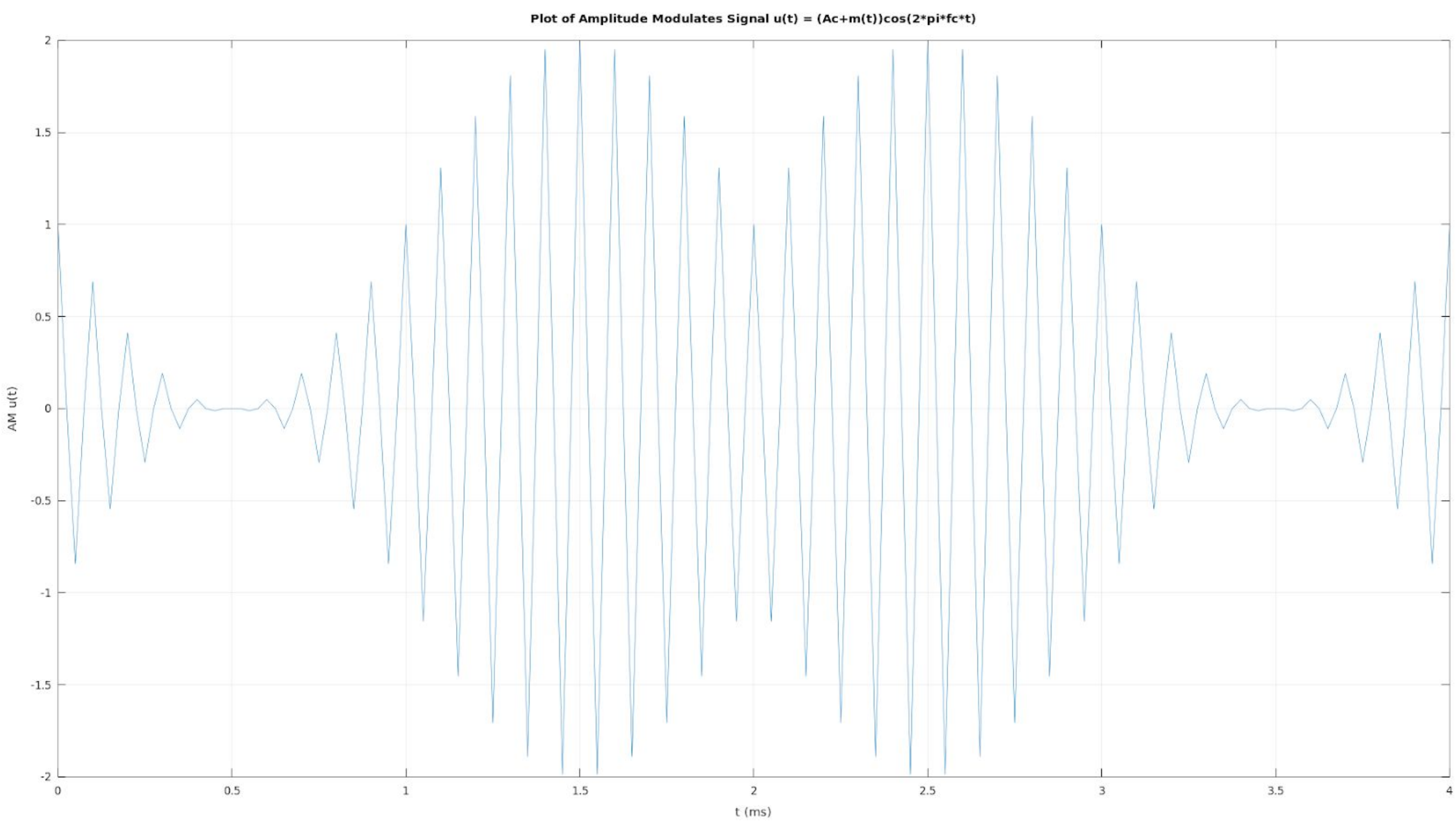
PSD of DSB of Message Signal



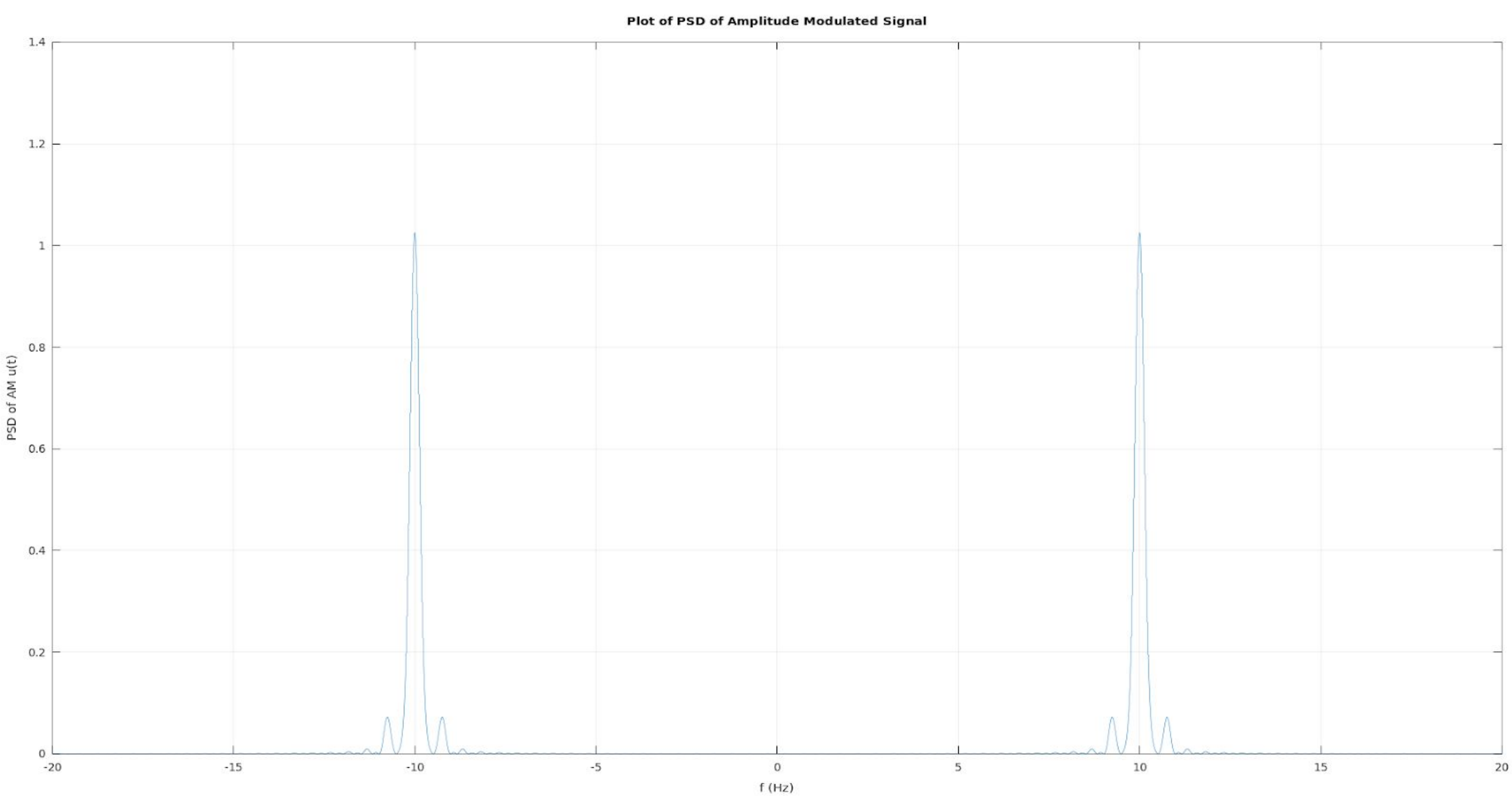


Relationship between PSD of message signal and PSD of its DSB signal is that the PSD of the message signal is shifted by ‘fc=10’ in the PSD of the DSB because of we multiple cos(2*pi*fc*t) to the message signal creating impulse two impulses in the frequency domain at ‘fc=10’ causing the shift. FFT of DSP makes the amplitude half and as we are plotting the square of the absolute value, the amplitude becomes one-fourth.

Amplitude Modulated Signal



PSD of Amplitude Modulated Signal



```
function question()
    m=16; %sampling rate as multiple of symbol rate
    T=1 % Symbol time period
    ns=4
    symbols = [-1;1;1;-1]
    [msg,time_msg] = msg_signal(m, symbols)
    figure(1,"position",[0,0,2000,1000])
    plot(time_msg,msg);
    title('Plot of message signa m(t)')
    xlabel("t (ms)");
    ylabel('m(t)');
    grid on
    print( 'fig1.png', '-dpngcairo','-S1800,1000', '-color' )
    pause;

    [msg_f,freqs] = msg_fft(msg,m)
    figure(2,"position",[0,0,2000,1000])
    subplot(2,1,1)
    plot(freqs,(abs(msg_f).^2)/(ns*T));
    title('Plot of Power Spectral Desity of m(t)')
    xlabel("f (Hz)");
    ylabel('PSD of m(t)');
    grid on
    [msg_avg,msgf_avg] = psd_u(m,symbols,T)
    subplot(2,1,2)
    plot(msgf_avg,msg_avg);
    title('Plot of Power Spectral Desity of m(t) over 24 runs')
    xlabel("f (Hz)");
    ylabel('PSD of m(t)');
    grid on
    print( 'fig2.png', '-dpngcairo','-S1800,1000', '-color' )
    pause;


    [msg_dsb,msgt_dsb] = msg_dsb(T, m, symbols)
    figure(3,"position",[0,0,1800,1000])
    plot(msgt_dsb,msg_dsb);
    title('Plot of DSB signal u(t) = m(t)*cos(2*pi*fc*t)')
    xlabel("t (ms)");
    ylabel('DSB m(t)');
    grid on
    print( 'fig3.png', '-dpngcairo','-S1800,1000', '-color' )
    pause;

    fs = 4*(10/T)
    [msg_dsb_f,dsb_freqs] = msg_fft(msg_dsb,fs)
    figure(4,"position",[0,0,2000,1000])
    subplot(2,1,1)
    plot(dsb_freqs,(abs(msg_dsb_f).^2)/(ns*T));
    title('Plot of Power Spectral Desity of DSB of m(t)')
    xlabel("f (Hz)");
    ylabel('PSD of DSB of m(t)');
    grid on
    [msg_dsb_avg,msg_f_dsb_avg] = psd_dsb(fs,symbols,T)
    subplot(2,1,2)
    plot(msg_f_dsb_avg,msg_dsb_avg);
    title('Plot of Power Spectral Desity of DSB of m(t) over 24 runs')
    xlabel("f (Hz)");
    ylabel('PSD of DSB of m(t)');
    grid on
    print( 'fig4.png', '-dpngcairo','-S1800,1000', '-color' )
    pause;

    figure(5,"position",[0,0,1800,1000])
    [msg_am,msg_t_am] = msg_am(msg_dsb, msgt_dsb, T)
    plot(msg_t_am,msg_am);
    title('Plot of Amplitude Modulates Signal u(t) = (Ac+m(t))cos(2*pi*fc*t)')
    xlabel("t (ms)");
    ylabel('AM u(t)');
    grid on
    print( 'fig5.png', '-dpngcairo','-S1800,1000', '-color' )
    pause;

    figure(6,"position",[0,0,1800,1000])
    [msg_am_f,am_freqs] = msg_fft(msg_am,fs)
    plot(am_freqs,(abs(msg_am_f).^2)/(ns*T));
    title('Plot of PSD of Amplitude Modulated Signal')
    xlabel("f (Hz)");
    ylabel('PSD of AM u(t)');
    grid on
    print( 'fig6.png', '-dpngcairo','-S1800,1000', '-color' )
    pause;

end
```

```
function [u,time_u] = msg_signal(m,symbols)
    %discrete time representation of sine pulse
    time_p = 0:1/m:1; %sampling times over duration of pulse
    p = sin(pi*time_p); %samples of the pulse

    %symbols to be modulated
    % symbols = [-1;1;1;-1]
    %UPSAMPLE BY m
    nsymbols = length(symbols);%length of original symbol sequence
    nsymbols_upsampled = 1+(nsymbols-1)*m;%length of upsampled symbol sequence
    symbols_upsampled = zeros(nsymbols_upsampled,1);%
    symbols_upsampled(1:m:nsymbols_upsampled)=symbols;%insert symbols with spacing M
    %GENERATE MODULATED SIGNAL BY DISCRETE TIME CONVOLUTION
    u=conv(symbols_upsampled,p);
    %PLOT MODULATED SIGNAL
    time_u = 0:1/m:(length(u)-1)/m; %unit of time = symbol time T
end

function [signal_freqdomain_centered,freqs] = msg_fft(u,m)
    ts=1/m; %sampling interval
    time_interval = 0:ts:1; %sampling time instants
    %%time domain signal evaluated at sampling instants
    signal_timedomain = sin(pi*time_interval); %sinusoidal pulse in our example
    fs_desired = 1/1000; %desired frequency granularity
    Nmin = ceil(1/(fs_desired*ts)); %minimum length DFT for desired frequency granularity
    %for efficient computation, choose FFT size to be power of 2
    Nfft = 2^(nextpow2(Nmin)) %FFT size = the next power of 2 at least as big as Nmin
    %Alternatively, one could also use DFT size equal to the minimum length
    %Nfft=Nmin;
    %note: fft function in Matlab is just the DFT when Nfft is not a power of 2
    %freq domain signal computed using DFT
    %fft function of size Nfft automatically zeropads as needed
    signal_freqdomain = ts*fft(u,Nfft);
    %fftshift function shifts DC to center of spectrum
    signal_freqdomain_centered = fftshift(signal_freqdomain);
    fs=1/(Nfft*ts); %actual frequency resolution attained
    %set of frequencies for which Fourier transform has been computed using DFT
    freqs = ((1:Nfft)-1-Nfft/2)*fs;
    %plot the magnitude spectrum
end
```

```
function [u_dsb,tu_dsb] = msg_dsb(T, m, symbols)
    [new_u,new_ut] = msg_signal(4*(10/T),symbols);
    u_dsb = new_u.*transpose(cos(2*pi*(10/T)*new_ut));
    tu_dsb = new_ut;
end

function [u_am,ut_am] = msg_am(u, ut, T)
    Ac = 1;
    u_am = u + (Ac).*transpose(cos(2*pi*(10/T)*ut));
    ut_am = ut;
end

function [u_avg,uf_avg] = psd_u(m,symbols,T)
    perm_symbols = perms(symbols)
    [u,ut] = msg_signal(m,symbols)
    [u_f,u_ff] = msg_fft(u,m)
    u_avg = zeros(length(u_f),1)
    for i = 1:1:24
        [ui, uti] = msg_signal(m,perm_symbols(i,:))
        [uif,uiff] = msg_fft(ui,m)
        u_avg = u_avg + ((abs(uif).^2)/(4*T))
    endfor
    u_avg = u_avg/24
    uf_avg = u_ff
end

function [dsb_avg,dsbf_avg] = psd_dsb(m,symbols,T)
    perm_symbols = perms(symbols)
    [u,ut] = msg_signal(m,symbols)
    [u_dsb,tu_dsb] = msg_dsb(T, m, symbols)
    [u_dsb,uf_dsb] = msg_fft(u_dsb,m)
    dsb_avg = zeros(length(uf_dsb),1)
    for i = 1:1:24
        [ui_dsb,tui_dsb] = msg_dsb(T, m, perm_symbols(i,:))
        [uif,uiff] = msg_fft(ui_dsb,m)
        dsb_avg = dsb_avg + ((abs(uif).^2)/(4*T))
    endfor
    dsb_avg = dsb_avg/24
    dsbf_avg = uf_dsb
end
```