ILLUSTRATIVE PROBLEM

Illustrative Problem 3.4 [Single-sideband example] The message signal

$$m(t) = \begin{cases} 1, & 0 \le t < \frac{t_0}{3} \\ -2, & \frac{t_0}{3} \le t < \frac{2t_0}{3} \\ 0, & \text{otherwise} \end{cases}$$

modulates the carrier $c(t) = \cos(2\pi f_c t)$ using an LSSB-AM scheme. It is assumed that $t_0 = 0.15$ s and $f_c = 250$ Hz.

1. Plot the Hilbert transform of the message signal and the modulated signal $u(t)$. Also plot the spectrum of the modulated signal.

2. Assuming the message signal is periodic with period t_0, determine the power in the modulated signal.

3. If a noise is added to the modulated signal such that the SNR after demodulation is 10 dB, determine the power in the noise.

SOLUTION

1. The Hilbert transform of the message signal can be computed using the Hilbert transform m-file of MATLAB—that is, hilbert.m. It should be noted, however, that this function returns a complex sequence whose real part is the original signal and whose

imaginary part is the desired Hilbert transform. Therefore, the Hilbert transform of the sequence m is obtained by using the command imag(hilbert(m)). Now, using the relation

$$u(t) = m(t)\cos(2\pi f_c t) + \hat{m}(t)\sin(2\pi f_c t) \tag{3.2.27}$$

we can find the modulated signal. Plots of $\hat{m}(t)$ and the spectrum of the LSSB-AM modulated signal $u(t)$ are shown in Figure 3.8.

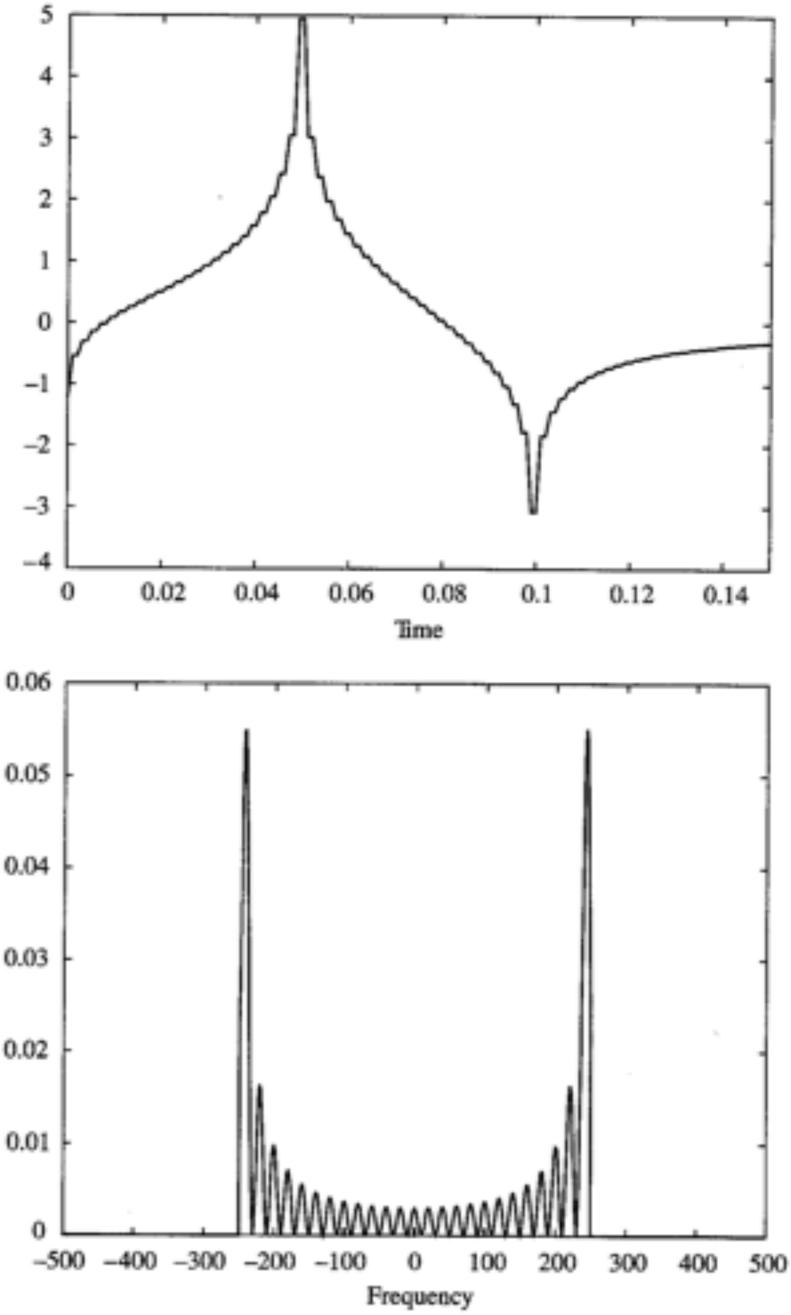

Figure 3.8 Hilbert transform and the spectrum of the LSSB-AM modulated signal for $m(t)$

2. The power in the message signal is

$$P_m = \frac{1}{0.15} \int_0^{0.15} m^2(t)\, dt = 1.667$$

and therefore

$$P_u = \frac{A_c^2}{4} P_m = 0.416$$

3. The post-demodulation SNR is given by

$$10 \log_{10} \left(\frac{P_R}{P_n} \right)_o = 10$$

Hence, $P_n = 0.1 P_R = 0.1 P_u = 0.0416$.

The MATLAB script for this problem follows.

M-FILE

```
% lssb.m
% Matlab demonstration script for LSSB-AM modulation. The message signal
% is +1 for 0 < t < t0/3, -2 for t0/3 < t < 2t0/3, and zero otherwise.
echo on
t0=.15;                                   % signal duration
ts=0.001;                                 % sampling interval
fc=250;                                   % carrier frequency
snr=10;                                   % SNR in dB (logarithmic)
fs=1/ts;                                  % sampling frequency
df=0.25;                                  % desired freq. resolution
t=[0:ts:t0];                              % time vector
snr_lin=10^(snr/10);                      % SNR
% the message vector
m=[ones(1,t0/(3*ts)),-2*ones(1,t0/(3*ts)),zeros(1,t0/(3*ts)+1)];
c=cos(2*pi*fc.*t);                        % carrier vector
udsb=m.*c;                                % DSB modulated signal
[UDSB,udssb,df1]=fftseq(udsb,ts,df);      % Fourier transform
UDSB=UDSB/fs;                             % scaling
f=[0:df1:df1*(length(udssb)-1)]-fs/2;     % frequency vector
n2=ceil(fc/df1);                          % location of carrier in freq. vector
% remove the upper sideband from DSB
UDSB(n2:length(UDSB)-n2)=zeros(size(UDSB(n2:length(UDSB)-n2)));
ULSSB=UDSB;                               % generate LSSB-AM spectrum
[M,m,df1]=fftseq(m,ts,df);                % Fourier transform
M=M/fs;                                   % scaling
u=real(ifft(ULSSB))*fs;                   % generate LSSB signal from spectrum
signal_power=spower(udsb(1:length(t)))/2;
                                          % compute signal power
%
```

```
noise_power=signal_power/snr_lin;          % compute noise power
noise_std=sqrt(noise_power);               % compute noise standard deviation
noise=noise_std*randn(1,length(u));        % generate noise vector
r=u+noise;                                 % add the signal to noise
[R,r,df1]=fftseq(r,ts,df);                 % Fourier transform
R=R/fs;                                    % scaling
pause  % Press a key to show the modulated signal power
signal_power
pause  % Press any key to see a plot of the message signal
clf
subplot(2,1,1)
plot(t,m(1:length(t)))
axis([0,0.15,-2.1,2.1])
xlabel('Time')
title('The message signal')
pause  % Press any key to see a plot of the carrier
subplot(2,1,2)
plot(t,c(1:length(t)))
xlabel('Time')
title('The carrier')
pause  % Press any key to see a plot of the modulated signal and its spectrum
clf
subplot(2,1,1)
plot([0:ts:ts*(length(u)-1)/8],u(1:length(u)/8))
xlabel('Time')
title('The LSSB-AM modulated signal')
subplot(2,1,2)
plot(f,abs(fftshift(ULSSB)))
xlabel('Frequency')
title('Spectrum of the LSSB-AM modulated signal')
pause   % Press any key to see the spectra of the message and the modulated signals
clf
subplot(2,1,1)
plot(f,abs(fftshift(M)))
xlabel('Frequency')
title('Spectrum of the message signal')
subplot(2,1,2)
plot(f,abs(fftshift(ULSSB)))
xlabel('Frequency')
title('Spectrum of the LSSB-AM modulated signal')

pause  % Press any key to see a noise sample
subplot(2,1,1)
plot(t,noise(1:length(t)))
title('Noise sample')
xlabel('Time')
pause  % Press a key to see the modulated signal and noise
subplot(2,1,2)
plot(t,r(1:length(t)))
title('Modulated signal and noise')
xlabel('Time')
subplot(2,1,1)
pause % Press any key to see the spectrum of the modulated signal
plot(f,abs(fftshift(ULSSB)))
```

```
title('Modulated signal spectrum')
xlabel('Frequency')
subplot(2,1,2)


pause  % Press a key to see the modulated signal noise in freq. domain
plot(f,abs(fftshift(R)))
title('Modulated signal noise spectrum')
xlabel('Frequency')
```

The m-files ussb_mod.m and lssb_mod.m given next modulate the message signal given in vector m using USSB and LSSB modulation schemes.

M-FILE

```
function u=ussb_mod(m,ts,fc)
%              u=ussb_mod(m,ts,fc)
%USSB_MOD      takes signal m sampled at ts and carrier
%              freq. fc as input and returns the USSB modulated
%              signal. ts << 1/2fc.
t=[0:length(m)-1]*ts;
u=m.*cos(2*pi*t)-imag(hilbert(m)).*sin(2*pi*t);
```

M-FILE

```
function u=lssb_mod(m,ts,fc)
%              u=lssb_mod(m,ts,fc)
%LSSB_MOD      takes signal m sampled at ts and carrier
%              freq. fc as input and returns the LSSB modulated
%              signal. ts << 1/2fc.
t=[0:length(m)-1]*ts;
u=m.*cos(2*pi*t)+imag(hilbert(m)).*sin(2*pi*t);
```