Snapshot of Communication Toolbox for Scilab

Vinayak Sharma and Mukesh Mehta

Department of Electrical Engineering
Indian Institute of Technology, Bombay

2nd December, 2010

Outline

Outline

1. BER for AWGN channel
2. Noise Bandwidth Calculation
3. Analog Modulation Schemes
4. Digital Modulation Schemes

Calculating BER using findber function

- Finds BER for channel having white gaussian noise.
- Can be used with pam,psk,qam,dpsk and several other schemes.
- Taking Bit energy to noise ratio as EbNo we can find BER as : $BER = findber(EbNo, Modulation, M)$
- Uses error expressions such as :

$$BER_{pam} = 2\frac{(M-1)}{kM} Q\left(\sqrt{\left(\frac{6log_2(M-1)EbNo}{M^2-1}\right)} \right)$$

$$BER_{psk} = \frac{2}{k} Q(\sqrt{2kEbNo}\sin(\frac{\pi}{M}))$$

BER for pulse amplitude modulation

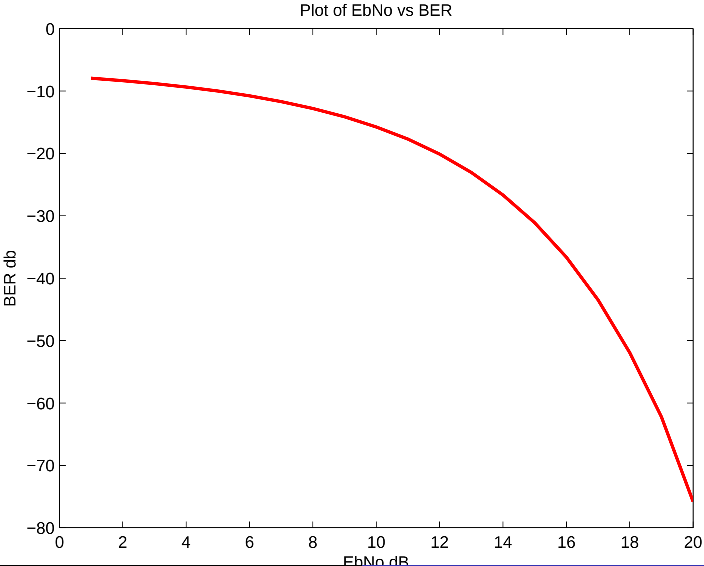
1. Define array EbNo,containing signal energy to noise ratio in decibels.
2. Define number of symbols M...in general power of two as 2,4,8...
3. Use findber(EbNo,'pam',M)
4. Plot the curve between BER and EbNo values using berplot(BER,EbNo)

Please open findber1.sce from folder COMM-SESSION in Scilab and observe the code.

Scilab Code:

- EbNo=1:20;// energy to noise ratio in decibels.
- M=8;// number of symbols M
- BER=findber(EbNo,'pam',8);
- plotber(EbNo,BER);// plot the curve between BER and EbNo

Execute this code using CTRL + E or CTRL + L.

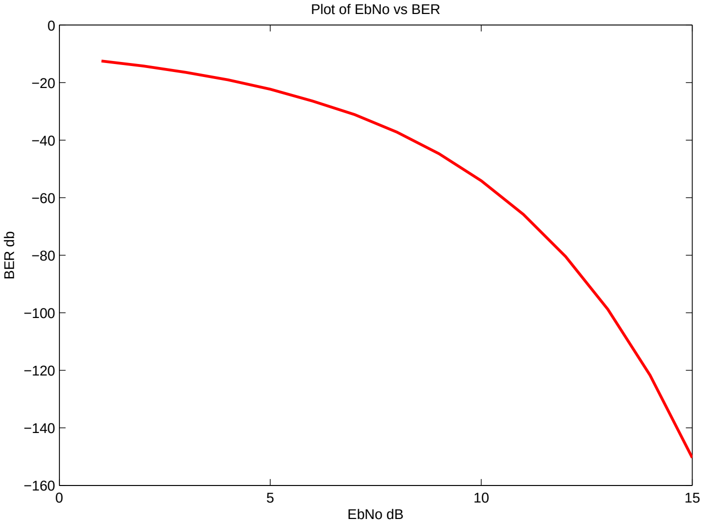

Plot of EbNo vs BER
BER db
EbNo dB
0
−10
−20
−30
−40
−50
−60
−70
−80
0
2
4
6
8
10
12
14
16
18
20

Similarly finding BER for phase shift keying

Please open findber2.sce from folder COMM-SESSION in Scilab and observe the code.

- EbNo=1:15;// energy to noise ratio in decibels.
- M=4;// number of symbols M
- BER=findber(EbNo,'psk',M,'nondiff');// using nondiff.encoding
- plotber(EbNo,BER);// plot the curve

Execute using CTRL + E or CTRL + L


Plot of EbNo vs BER
0
−20
−40
−60
−80
−100
−120
−140
−160
BER db
0
5
10
15
EbNo dB

Try this yourself....

Find BER for differential phase shift keying using these steps :

1. Define array EbNo to have values 0.1,0.2,0.3....such values upto 3.
2. Define number of symbols M = 4
3. Use findber(EbNo,'dpsk',M)
4. Plot the curve between BER and EbNo values using berplot

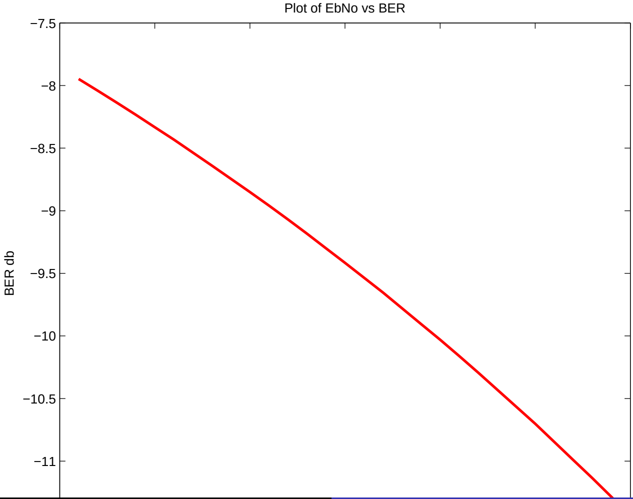
Open a new file and code as described...then save it as findber11.sce
Now execute using CTRL + E or CTRL + L.

Here is the code...

- EbNo=0.1:0.1:3;
- M=4;
- BER=findber(EbNo,'dpsk',M);
- plotber(EbNo,BER); // plot the curve

The response is close to linear in this case

Plot between BER dB(on y axis) and EbNo dB(on x axis)



Outline

Using bandwidthn function

NoiseBW = bandwidthn(Nsample,freqsampl,b,a)
where :

1. Nsample is the number of samples to be used in computation
2. freqsampl is the sampling frequency for the system
3. b and a are the arrays having numerator and denominator coefficients of digital filter.

$$H(z) = \frac{b_1 + b_2 z^{-1} + b_3 z^{-2} \ldots\ldots}{a_1 + a_2 z^{-1} + a_3 z^{-2} \ldots\ldots}$$

$$Here \ \ b = [b_1, b_2, b_3 \ldots]$$

$$a = [a_1, a_2, a_3 \ldots]$$

1. Let us find noise bandwidth for this filter :

$$H(z) = \frac{-1-2.z^{-1}-z^{-2}}{2+3.z^{-1}-2.z^{-2}+Z^{-3}}$$

2. Supply the values of b and a for this filter.
3. Take Nsample=50;
4. Select sampling frequency Fs=20;

Please open noisebandwidth1.sce from folder COMM-SESSION in Scilab and observe the code.

Scilab code is :

- b = [-1, -2, -1];
- a= [2, 3, -2, 1];
- Nsample=50;
- Fs=20;
- B=bandwidthn(Nsample,Fs,b,a);
- disp('Hz',B,'noise bandwidth : ');

Execute using CTRL + E or CTRL + L.

Try this yourself...

1. Find noise bandwidth for this filter :

   $$H(z) = \frac{1+4.z^{-1}+2.z^{-2}}{3-2.z^{-1}+z^{-2}+4.Z^{-3}}$$

2. Supply the values of b and a for this filter.
3. Take desired number of samples...say Nsample=200;
4. Select sampling frequency Fs=40;

Open a new file and code as described...then save it as noisebandwidth11.sce
Now execute using CTRL + E or CTRL + L.

Scilab code :

- b = [1, 4, 2];
- a= [3, -2, 1, 4];
- Nsample=200;
- Fs=40;
- B=bandwidthn(Nsample,Fs,b,a);
- disp('Hz',B,'noise bandwidth : ');

Ensure that given filters qualify as low pass filters.
Let us use some other type of filter.
For example replace array b with b = [-1 4 -2 -1] in last example and execute the code.
(If you could not do it, please open the code noisebandwidth2.sce in COMM-SESSION folder.)
Do we still get the result ??

Outline

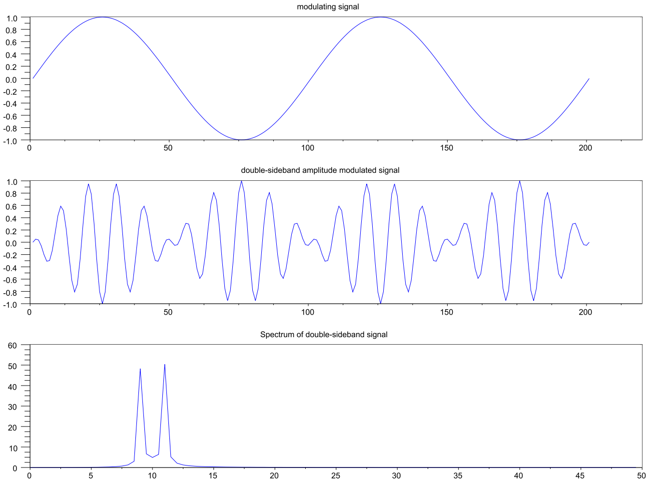
Amplitude Modulation

Modulating a sinusoidal baseband signal x(t) with carrier of frequency Fc :

- Select sampling frequency Fs
- Generate sampling instants..say from 0 to 2 seconds
- Set the carrier initial phase
- Set the carrier frequency Fc and amplitude
- Supply message signal
- Modulate it using ampmod function
- Find the spectrum using fft.

Scilab code

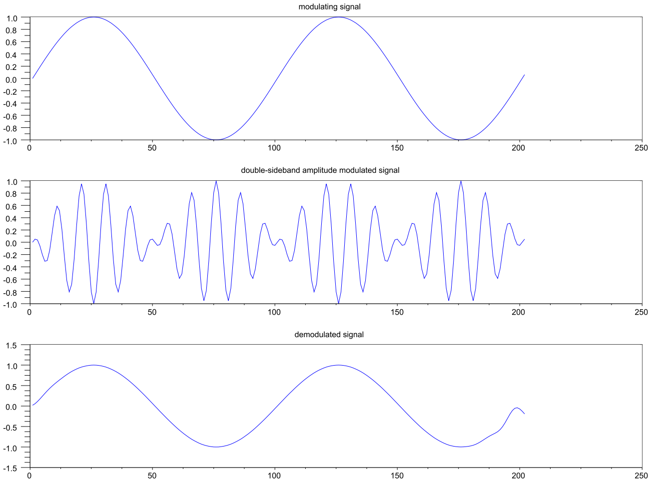
- Fs =100; //Sampling Frequency
- $t = [0 : 2 * Fs]'/Fs$; // sampling instants
- $iniphase = 0$; //initial carrier phase
- $Fc = 10$; // Carrier frequency
- $carramp = 0$; //carrier amplitude
- $x = sin(2 * \%pi * t)$; //Sinusoidal signal
- $y = ampmod(x, Fc, Fs, iniphase, carramp)$;
- $z = fft(y)$; // find frequency spectrum
- $zz = abs(z(1 : length(z)/2))$; //take one of the sidebands
- $axis = (0 : Fs/length(zz) : Fs - (Fs/length(zz)))/2$; // frequency axis
- subplot(3,1,1); plot(x); // plot message signal
- subplot(3,1,2); plot(y); // plot the modulated signal
- subplot(3,1,3); plot(axis,zz); // plot spectrum


modulating signal
1.0
0.8
0.6
0.4
0.2
0.0
-0.2
-0.4
-0.6
-0.8
-1.0
0
50
100
150
200
double-sideband amplitude modulated signal
1.0
0.8
0.6
0.4
0.2
0.0
-0.2
-0.4
-0.6
-0.8
-1.0
0
50
100
150
200
Spectrum of double-sideband signal
60
50
40
30
20
10
0
0
5
10
15
20
25
30
35
40
45
50

Demodulation

Now Let us demodulate the modulated signal y to get back our baseband sigal x.
Use all the same parameters as in the modulation.

- $yy = ampdemod(y, Fc, Fs, iniphase, carramp)$;
- subplot(3,1,1); plot(x); // plot message signal
- subplot(3,1,2); plot(y); // plot the modulated signal
- subplot(3,1,3); plot(yy); // plot demodulated output


modulating signal
double-sideband amplitude modulated signal
demodulated signal
1.0
0.8
0.6
0.4
0.2
0.0
-0.2
-0.4
-0.6
-0.8
-1.0
1.5
0.5
-0.5
-1.5
0
50
100
150
200
250

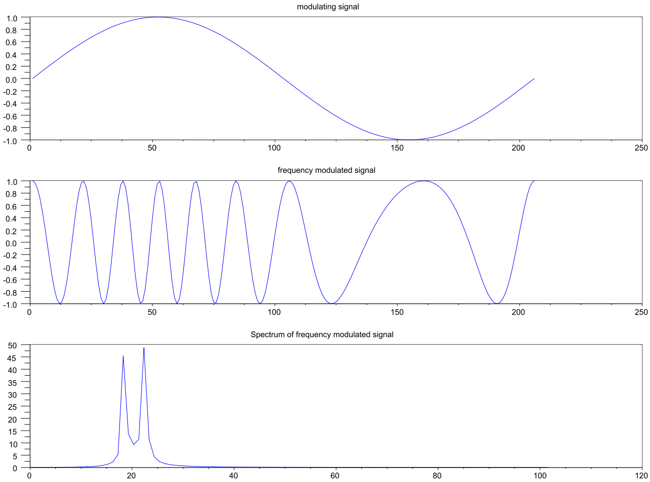
Frequency Modulation

Given a sinusoidal signal x(t), Let us frequency modulate it using carrier with frequency Fc:

- Select sampling frequency, instants, carrier phase and frequency as in the last example.
- Select frequency deviation for FM
- Now Modulate sampled version of x(t) using freqmod function
- Find the spectrum using fft.

Scilab code

- Fs =200; //Sampling Frequency
- $t = [0 : Fs]'/Fs$; // sampling instants
- $iniphase = 0$; //initial carrier phase
- $freqdev = 6$;
- $Fc = 8$; // Carrier frequency
- $x = sin(2 * \%pi * t)$; //Sinusoidal signal
- $y = freqmod(x, Fc, Fs, freqdev, iniphase)$;
- $z = fft(y)$; // find frequency spectrun
- $zz = abs(z(1 : length(z)/2))$; //take positive frequencies
- $axis = (0 : Fs/length(zz) : Fs - (Fs/length(zz)))/2$; // frequency axis
- subplot(3,1,1); plot(x); // plot message signal
- subplot(3,1,2); plot(y); // plot the modulated signal
- subplot(3,1,3); plot(axis,zz); // plot spectrum


modulating signal
frequency modulated signal
Spectrum of frequency modulated signal

Demodulation

Demodulate the modulated signal y to get back our baseband sigal x.

Use all the same parameters as in the modulation.

- $yy = fmdemod(y, Fc, Fs, freqdev, iniphase);$
- subplot(3,1,1); plot(x); // plot message signal
- subplot(3,1,2); plot(y); // plot the modulated signal
- subplot(3,1,3); plot(yy); // plot demodulated output

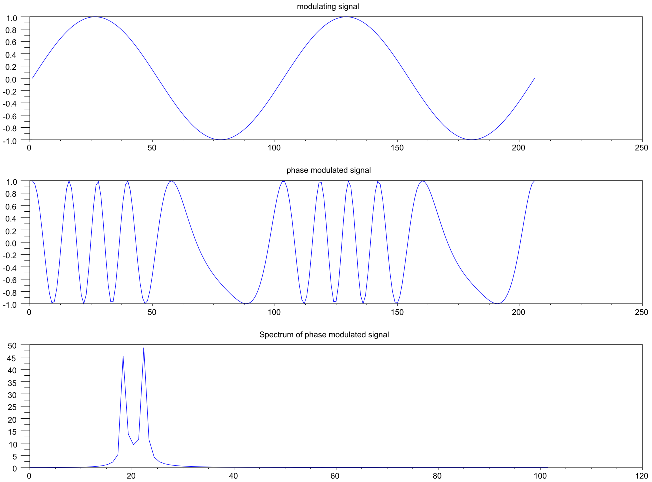
Phase Modulation

Used in the same way as the frequency modulation, the function used here is phasemod.

- Select sampling frequency, instants, carrier phase and frequency.
- Select phase deviation here.
- Modulate sampled version of x(t) using phasemod
- Find the spectrum using fft.

Scilab code

- Fs =200; //Sampling Frequency
- $t = [0 : Fs]'/Fs$; // sampling instants
- $iniphase = 0$; //initial carrier phase
- $phasedev = 6$;
- $Fc = 8$; // Carrier frequency
- $x = sin(2 * \%pi * t)$; //Sinusoidal signal
- $y = phasemod(x, Fc, Fs, phasedev, iniphase)$;
- $z = fft(y)$; // find frequency spectrun
- $zz = abs(z(1 : length(z)/2))$; //take positive frequencies
- $axis = (0 : Fs/length(zz) : Fs - (Fs/length(zz)))/2$; // frequency axis
- subplot(3,1,1); plot(x); // plot message signal
- subplot(3,1,2); plot(y); // plot the modulated signal
- subplot(3,1,3); plot(axis,zz); // plot spectrum


modulating signal
1.0
0.8
0.6
0.4
0.2
0.0
-0.2
-0.4
-0.6
-0.8
-1.0
0
50
100
150
200
250
phase modulated signal
1.0
0.8
0.6
0.4
0.2
0.0
-0.2
-0.4
-0.6
-0.8
-1.0
0
50
100
150
200
250
Spectrum of phase modulated signal
50
45
40
35
30
25
20
15
10
5
0
0
20
40
60
80
100
120

Demodulation

Using the same parameters as in modulation.

- $yy = phasedemod(y, Fc, Fs, phasedev, iniphase)$;
- subplot(3,1,1); plot(x); // plot message signal
- subplot(3,1,2); plot(y); // plot the modulated signal
- subplot(3,1,3); plot(yy); // plot demodulated output

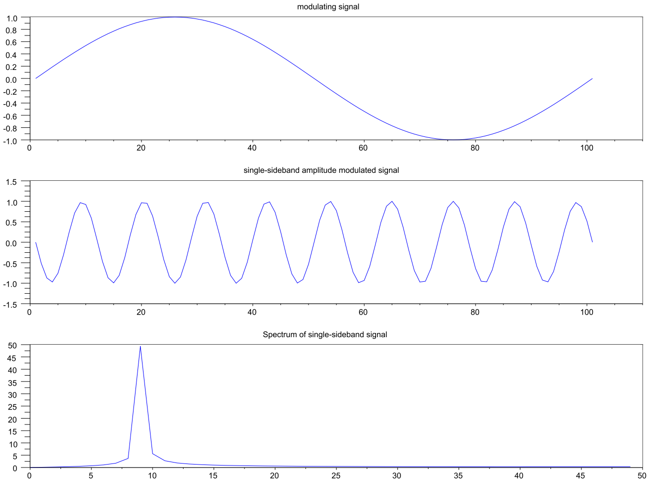
Single Sideband Modulation

Used in the same way as the frequency modulation, the function used here is phasemod.

- Select sampling frequency, instants, carrier phase and frequency.
- Modulate sampled version of x(t) using ssbampmod
- Find the spectrum using fft.

Scilab code

- Fs =100; //Sampling Frequency
- $t = [0 : Fs]'/Fs$; // sampling instants
- $iniphase = 0$; //initial carrier phase
- $Fc = 10$; // Carrier frequency
- $x = sin(2 * \%pi * t)$; //Sinusoidal signal
- $y = ssbampmod(x, Fc, Fs, iniphase,' lower')$;
- $z = fft(y)$; // find frequency spectrun
- $zz = abs(z(1 : length(z)/2))$; //take positive frequencies
- $axis = (0 : Fs/length(zz) : Fs - (Fs/length(zz)))/2$; // frequency axis
- subplot(3,1,1); plot(x); // plot message signal
- subplot(3,1,2); plot(y); // plot the modulated signal
- subplot(3,1,3); plot(axis,zz); // plot spectrum


modulating signal
1.0
0.8
0.6
0.4
0.2
0.0
-0.2
-0.4
-0.6
-0.8
-1.0
0
20
40
60
80
100
single-sideband amplitude modulated signal
1.5
1.0
0.5
0.0
-0.5
-1.0
-1.5
0
20
40
60
80
100
Spectrum of single-sideband signal
50
45
40
35
30
25
20
15
10
5
0
0
5
10
15
20
25
30
35
40
45
50

Demodulation

Using the same parameters as in modulation.

- $yy = ssbampdemod(y, Fc, Fs, iniphase)$;
- subplot(3,1,1); plot(x); // plot message signal
- subplot(3,1,2); plot(y); // plot the modulated signal
- subplot(3,1,3); plot(yy); // plot demodulated output

Outline

Phase Shift Keying Modulation

- Select number of symbols M.
- Supply the data X in range from 0 to M-1 to be modulated
- Supply initial phase for the carrier
- Use pskmdl to modulate X.
- Gray encoding is used for X

Scilab code

- M = 4;
- $X = [3, 2, 1, 2, 1, 0, 1]$;
- $iniphase = \%pi/4$;
- $Y = pskmdl(X, M, iniphase, "gray")$;

Demodulate Y using pskdemdl as :

- $YY = pskdemdl(Y, M, iniphase, "gray")$;

Pulse Amplitude Modulation

- Select number of symbols M.
- Supply the data X in range from 0 to M-1 to be modulated
- Supply initial phase for the carrier
- Use pammdl to modulate X.
- Gray encoding is used for X

Scilab code

- M = 4;
- $X = [1, 2, 0, 3, 1, 0];$
- $iniphase = \%pi/2;$
- $Y = pammdl(X, M, iniphase, "gray");$

Demodulate Y using pskdemdl as :

- $YY = pamdemdl(Y, M, iniphase, "gray");$

Very Soon in Scilab...

- Functions for encoding and decoding such as huffman and BCH codes.
- Functions for modelling rayleigh-rician channel models.
- Functions for Galois Field computations.

...Communication toolbox will have all this features and more.

- Simon, M. K., Alouini, M. S., Digital Communication over Fading Channels A Unified Approach to Performance Analysis, 1st ed., Wiley, 2000.
- Proakis, J. G., Digital Communications, 4th ed., McGraw-Hill, 2001

Thank You !