Lab 6

Aravind Reddy V

IMT 2015 524

Question 1

I and Q component of FM signal, kf = 0.25

|  |
| --- |
| Fig. 1 – I and Q components, theta/pi of FM signal |

Phase decreases when message symbol is negative and increases when it is positive

When the message signal changes, there is a phase change in either I or Q component.

Question 2

I and Q component of FM signal, kf = 4

|  |
| --- |
| Fig.2 – kf = 4 |

Patterns in I and Q are more difficult to observe now

Phase, however, can be read with the same ease.

Question 3

|  |
| --- |
| ../../Downloads/new%20doc%202017-10-27%2022.24.32_1.jpg  Fig.3 - derivation |

Question 4

|  |
| --- |
| 4a.png  Fig. 4a – Estimated signal, kf = 0.25 |

|  |
| --- |
| 4b.png  Fig. 4b – Estimated signal, kf = 4 |

From eyeballing, these are good estimates of the original message signal.

Question 5

|  |
| --- |
| 5/5a.png  Fig. 5a – Random phase added to complex envelope – Kf = 0.25 |

|  |
| --- |
| 5/5b.png  Fig. 5b – Random phase added to complex envelope – Kf = 4 |

Addition of phase does not produce any distortion.

Question 6

|  |
| --- |
| 6/6a.png  Fig. 6a – Random phase and frequency offset – Kf = 0.25 |

|  |
| --- |
| 6/6b.png  Fig. 6b - Random phase and frequency offset – Kf = 4 |

Frequency offset produces a DC shift in the estimated message

Question 7

|  |
| --- |
| 7/7a.png  Fig. 7a – PSD of FM signal. Kf = 0.25 |

|  |
| --- |
| 7/7b.png  Fig. 7b – PSD of FM signal. Kf = 4 |

The peak starts splitting as K increases, at kf = 0.5, the peak is completely split.

Question 8

No, they are not.

As seen from plots in the previous question,

for k = 0.25, bandwidth is approximately 0.75 Hz

for k = 0.40, bandwidth is approximately 4.00 Hz

From Carson’s formula,

for k = 0.25, bandwidth is approximately 2.50 Hz

for k = 0.40, bandwidth is approximately 10.0 Hz

|  |
| --- |
| 8/7a.png  Fig. 8a – PSD of message signal, kf = 0.25 – bandwidth = 1 Hz |

|  |
| --- |
| 8/7b.png  Fig. 8a – PSD of message signal, kf = 4 – bandwidth = 1 Hz |

Bandwidth calculation via Carson’s formula:

Message bandwidth B = 1 Hz

Beta = kf \* B

Bandwidth = 2B(1 + Beta)

So, for Kf = 0.25, B = 1 Hz, Beta = 0.25 and Bandwidth = 2.5 Hz

And for Kf = 4, B = 1 Hz, Beta = 4 and Bandwidth = 10 Hz

Question 9

|  |
| --- |
| 9/7a.png  Fig. 9a – PSD of FM for kf = 0.25 |

|  |
| --- |
| 9/7b.png  Fig. 9b – PSD of FM for Kf = 4 |

From these plots,

for k = 0.25, bandwidth is approximately 1 Hz

for k = 0.40, bandwidth is approximately 5 Hz

We know from previous calculations that message signals have bandwidths as follows

From Carson’s formula,

for k = 0.25, bandwidth is approximately 2.50 Hz

for k = 0.40, bandwidth is approximately 10.0 Hz

So no, the results are again not in consistence with Carson’s formula.

Question 10

|  |
| --- |
| 10/7a.png  Fig. 10a – PSD of FM with message bits containing numbers drawn from a Gaussian distribution with the same variance, kf = 0.25. Bandwidth = 1 Hz |

|  |
| --- |
| 10/7b.png  Fig. 10b - PSD of FM with message bits containing numbers drawn from a Gaussian distribution with the same variance, kf = 4. Bandwidth = 10 Hz |

Spectral occupancy in kf = 0.25 is almost the same

Spectral occupancy in kf = 4 is strikingly different, however. In the previous case, the distribution was concentrated within 5 Hz, but here it spreads till 10 Hz but with major part within 5 Hz.

Question 11

The bandwidth increases by 1000 because the unit of time in divided by 1000. So the bandwidth is

for k = 0.25, bandwidth is approximately 1 KHz

for k = 0.40, bandwidth is approximately 5 KHz

Question 12

|  |
| --- |
| 12/12a.png  Fig. 12a – FM signal. Kf = 0.25 |

|  |
| --- |
| 12/12b.png  Fig. 12b – FM signal. Kf = 4 |

Question 13

|  |
| --- |
| 13/13a.png  Fig. 13a – Demodulated FM signal |

Question 14

Randn returns a matrix with normally distributed random elements having zero mean and variance one.

Randsrc generates a random number between +1 and -1.

Codes:

Note: You can also download all files from this link: https://goo.gl/mPKhZP

1.

oversampling\_factor = 16;

%for a pulse with amplitude one, the max frequency deviation is given by kf

kf=0.25;

%increase the oversampling factor if kf (and hence frequency deviation, and hence bw of FM s

oversampling\_factor = ceil(max(kf,1)\*oversampling\_factor);

ts=1/oversampling\_factor;%sampling time

nsamples = ceil(1/ts);

pulse = ones(nsamples,1); %rectangular pulse

nsymbols =10;

symbols=zeros(nsymbols,1);

%random symbol sequence

symbols = sign(rand(nsymbols,1)-0.5);

symbols

%generate digitally modulated message

nsymbols\_upsampled=1+(nsymbols-1)\*nsamples;

symbols\_upsampled=zeros(nsymbols\_upsampled,1);

symbols\_upsampled(1:nsamples:nsymbols\_upsampled)=symbols;

message = conv(symbols\_upsampled,pulse);

%FM signal phase obtained by integrating the message

theta = 2\*pi\*kf\*ts\*cumsum(message);

cenvelope=exp(j\*theta);

L=length(cenvelope);

time=(0:L-1)\*ts;

Icomponent = real(cenvelope);

Qcomponent= imag(cenvelope);

subplot(4, 1, 1);

%plot Message

plot(time, message);

title(["Message signal. symbols = ", mat2str(symbols)]);

xlabel("time (s)");

ylabel("amplitude");

subplot(4, 1, 2);

%plot I component

plot(time,Icomponent);

title("I component");

xlabel("time (s)");

ylabel("amplitude");

subplot(4, 1, 3);

%plot Q component

plot(time, Qcomponent);

title("Q component");

xlabel("time (s)");

ylabel("amplitude");

subplot(4, 1, 4);

%plot theta

plot(time, theta./pi);

title(["\\theta", "(t)/", "\\pi"]);

xlabel("time (s)");

ylabel("amplitude");

print -dpng 1.png

2.

oversampling\_factor = 16;

%for a pulse with amplitude one, the max frequency deviation is given by kf

kf=4;

%increase the oversampling factor if kf (and hence frequency deviation, and hence bw of FM s

oversampling\_factor = ceil(max(kf,1)\*oversampling\_factor);

ts=1/oversampling\_factor;%sampling time

nsamples = ceil(1/ts);

pulse = ones(nsamples,1); %rectangular pulse

nsymbols =10;

symbols=zeros(nsymbols,1);

%random symbol sequence

symbols = sign(rand(nsymbols,1)-0.5);

symbols

%generate digitally modulated message

nsymbols\_upsampled=1+(nsymbols-1)\*nsamples;

symbols\_upsampled=zeros(nsymbols\_upsampled,1);

symbols\_upsampled(1:nsamples:nsymbols\_upsampled)=symbols;

message = conv(symbols\_upsampled,pulse);

%FM signal phase obtained by integrating the message

theta = 2\*pi\*kf\*ts\*cumsum(message);

cenvelope=exp(j\*theta);

L=length(cenvelope);

time=(0:L-1)\*ts;

Icomponent = real(cenvelope);

Qcomponent= imag(cenvelope);

subplot(4, 1, 1);

%plot Message

plot(time, message);

title(["Message signal. symbols = ", mat2str(symbols)]);

xlabel("time (s)");

ylabel("amplitude");

subplot(4, 1, 2);

%plot I component

plot(time,Icomponent);

title("I component");

xlabel("time (s)");

ylabel("amplitude");

subplot(4, 1, 3);

%plot Q component

plot(time, Qcomponent);

title("Q component");

xlabel("time (s)");

ylabel("amplitude");

subplot(4, 1, 4);

%plot theta

plot(time, theta./pi);

title(["\\theta", "(t)/", "\\pi"]);

xlabel("time (s)");

ylabel("amplitude");

print -dpng 2.png

4.

oversampling\_factor = 16;

%for a pulse with amplitude one, the max frequency deviation is given by kf

kf=0.25;

%increase the oversampling factor if kf (and hence frequency deviation, and hence bw of FM s

oversampling\_factor = ceil(max(kf,1)\*oversampling\_factor);

ts=1/oversampling\_factor;%sampling time

nsamples = ceil(1/ts);

pulse = ones(nsamples,1); %rectangular pulse

nsymbols =10;

symbols=zeros(nsymbols,1);

%random symbol sequence

symbols = sign(rand(nsymbols,1)-0.5);

symbols

%generate digitally modulated message

nsymbols\_upsampled=1+(nsymbols-1)\*nsamples;

symbols\_upsampled=zeros(nsymbols\_upsampled,1);

symbols\_upsampled(1:nsamples:nsymbols\_upsampled)=symbols;

message = conv(symbols\_upsampled,pulse);

%FM signal phase obtained by integrating the message

theta = 2\*pi\*kf\*ts\*cumsum(message);

cenvelope=exp(j\*theta);

L=length(cenvelope);

time=(0:L-1)\*ts;

Icomponent = real(cenvelope);

Qcomponent= imag(cenvelope);

% subplot(4, 1, 1);

% %plot Message

% plot(time, message);

% title(["Message signal. symbols = ", mat2str(symbols)]);

% xlabel("time (s)");

% ylabel("amplitude");

% subplot(4, 1, 2);

% %plot I component

% plot(time,Icomponent);

% title("I component");

% xlabel("time (s)");

% ylabel("amplitude");

% subplot(4, 1, 3);

% %plot Q component

% plot(time, Qcomponent);

% title("Q component");

% xlabel("time (s)");

% ylabel("amplitude");

% subplot(4, 1, 4);

% %plot theta

% plot(time, theta./pi);

% title(["\\theta", "(t)/", "\\pi"]);

% xlabel("time (s)");

% ylabel("amplitude");

% print -dpng 1.png

%baseband discriminator

%differencing operation approximates derivative

Iderivative = [0;diff(Icomponent)]/ts;

Qderivative = [0;diff(Qcomponent)]/ts;

message\_estimate = (1/(2\*pi\*kf))\*(Icomponent.\*Qderivative - Qcomponent.\*Iderivative)./(Icomponent.^2 .+ Qcomponent.^2);

subplot(2, 1, 1);

plot(time, message);

title(["Message signal. symbols = ", mat2str(symbols)]);

xlabel("time (s)");

ylabel("amplitude");

subplot(2, 1, 2);

plot(time, message);

title("Estimated message. kf = 0.25");

xlabel("time (s)");

ylabel("amplitude");

print -dpng 4a.png

5.

oversampling\_factor = 16;

%for a pulse with amplitude one, the max frequency deviation is given by kf

kf=0.25;

%increase the oversampling factor if kf (and hence frequency deviation, and hence bw of FM s

oversampling\_factor = ceil(max(kf,1)\*oversampling\_factor);

ts=1/oversampling\_factor;%sampling time

nsamples = ceil(1/ts);

pulse = ones(nsamples,1); %rectangular pulse

nsymbols =10;

symbols=zeros(nsymbols,1);

%random symbol sequence

symbols = sign(rand(nsymbols,1)-0.5);

symbols

%generate digitally modulated message

nsymbols\_upsampled=1+(nsymbols-1)\*nsamples;

symbols\_upsampled=zeros(nsymbols\_upsampled,1);

symbols\_upsampled(1:nsamples:nsymbols\_upsampled)=symbols;

message = conv(symbols\_upsampled,pulse);

%FM signal phase obtained by integrating the message

theta = 2\*pi\*kf\*ts\*cumsum(message);

cenvelope=exp(j\*theta);

phi = 2\*pi\*rand; %phase uniform over [0,2pi]

phi = 0

cenvelope = cenvelope.\*exp(j\*phi); % adding random phase to theta

% e^(x)\*e^(y) = e^(x+y)

%now apply baseband discriminator

L=length(cenvelope);

time=(0:L-1)\*ts;

Icomponent = real(cenvelope);

Qcomponent= imag(cenvelope);

%baseband discriminator

%differencing operation approximates derivative

Iderivative = [0;diff(Icomponent)]/ts;

Qderivative = [0;diff(Qcomponent)]/ts;

message\_estimate = (1/(2\*pi\*kf))\*(Icomponent.\*Qderivative - Qcomponent.\*Iderivative)./(Icomponent.^2 .+ Qcomponent.^2);

subplot(2, 1, 1);

plot(time, message);

title(["Message signal. symbols = ", mat2str(symbols)]);

xlabel("time (s)");

ylabel("amplitude");

subplot(2, 1, 2);

plot(time, message);

title("Estimated message. kf = 0.25");

xlabel("time (s)");

ylabel("amplitude");

print -dpng 5a.png

6.

oversampling\_factor = 16;

%for a pulse with amplitude one, the max frequency deviation is given by kf

kf=0.25;

%increase the oversampling factor if kf (and hence frequency deviation, and hence bw of FM s

oversampling\_factor = ceil(max(kf,1)\*oversampling\_factor);

ts=1/oversampling\_factor;%sampling time

nsamples = ceil(1/ts);

pulse = ones(nsamples,1); %rectangular pulse

nsymbols =10;

symbols=zeros(nsymbols,1);

%random symbol sequence

symbols = sign(rand(nsymbols,1)-0.5);

symbols

%generate digitally modulated message

nsymbols\_upsampled=1+(nsymbols-1)\*nsamples;

symbols\_upsampled=zeros(nsymbols\_upsampled,1);

symbols\_upsampled(1:nsamples:nsymbols\_upsampled)=symbols;

message = conv(symbols\_upsampled,pulse);

%FM signal phase obtained by integrating the message

theta = 2\*pi\*kf\*ts\*cumsum(message);

cenvelope=exp(j\*theta);

L=length(cenvelope);

time=(0:L-1)\*ts;

phi = 2\*pi\*rand; %phase uniform over [0,2 pi]

df = 0.3;

cenvelope = cenvelope.\*exp(j\*(2\*pi\*df\*time'+phi));

% adding random phase and frequency offset to theta

% e^(x)\*e^(y) = e^(x+y)

%now apply baseband discriminator

Icomponent = real(cenvelope);

Qcomponent= imag(cenvelope);

%baseband discriminator

%differencing operation approximates derivative

Iderivative = [0;diff(Icomponent)]/ts;

Qderivative = [0;diff(Qcomponent)]/ts;

message\_estimate = (1/(2\*pi\*kf))\*(Icomponent.\*Qderivative - Qcomponent.\*Iderivative)./(Icomponent.^2 .+ Qcomponent.^2);

subplot(2, 1, 1);

plot(time, message);

title(["Message signal. symbols = ", mat2str(symbols)]);

xlabel("time (s)");

ylabel("amplitude");

subplot(2, 1, 2);

plot(time, message\_estimate);

title("Estimated message. kf = 0.25");

xlabel("time (s)");

ylabel("amplitude");

print -dpng 6a.png

7.

oversampling\_factor = 16;

%for a pulse with amplitude one, the max frequency deviation is given by kf

kf=0.25;

%increase the oversampling factor if kf (and hence frequency deviation, and hence bw of FM s

oversampling\_factor = ceil(max(kf,1)\*oversampling\_factor);

ts=1/oversampling\_factor; %sampling time

nsamples = ceil(1/ts);

pulse = ones(nsamples,1); %rectangular pulse

% pulse\_time = 0:ts:1;

% pulse = sin(2\*pi\*pulse\_time);

% calculating PSD

nsymbols =1000;

symbols=zeros(nsymbols,1);

nruns=1000;

fs\_desired=0.1;

Nmin = ceil(1/(fs\_desired\*ts)); %minimum length DFT for desired frequency granularity

message\_length=1+(nsymbols-1)\*nsamples+length(pulse)-1;

Nmin = max(message\_length,Nmin);

% %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

psd=zeros(Nfft,1);

for runs=1:nruns,

%random symbol sequence

symbols = sign(rand(nsymbols,1)-0.5);

nsymbols\_upsampled = 1+(nsymbols-1)\*nsamples;

symbols\_upsampled = zeros(nsymbols\_upsampled,1);

symbols\_upsampled(1:nsamples:nsymbols\_upsampled) = symbols;

message = conv(symbols\_upsampled,pulse);

%FM signal phase

theta = 2\*pi\*kf\*ts\*cumsum(message);

cenvelope = exp(j\*theta);

time = (0:length(cenvelope)-1)\*ts;

% %freq domain signal computed using DFT

cenvelope\_freq = ts\*fft(cenvelope,Nfft); %FFT of size Nfft, automatically zeropads as needed

cenvelope\_freq\_centered = fftshift(cenvelope\_freq); %shifts DC to center of spectrum

psd=psd+abs(cenvelope\_freq\_centered).^2;

end

psd=psd/(nruns\*nsymbols);

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 PSD

plot(freqs,psd);

title(["PSD of fm signal. kf = ", num2str(kf)]);

ylabel("Magnitude");

xlabel("Frequency (Hz)");

xlim([-1.5, 1.5]);

print -dpng 7a.png

8.

oversampling\_factor = 16;

%for a pulse with amplitude one, the max frequency deviation is given by kf

kf=0.25;

%increase the oversampling factor if kf (and hence frequency deviation, and hence bw of FM s

oversampling\_factor = ceil(max(kf,1)\*oversampling\_factor);

ts=1/oversampling\_factor; %sampling time

nsamples = ceil(1/ts);

pulse = ones(nsamples,1); %rectangular pulse

% calculating PSD

nsymbols =1000;

symbols=zeros(nsymbols,1);

nruns=1000;

fs\_desired=0.1;

Nmin = ceil(1/(fs\_desired\*ts)); %minimum length DFT for desired frequency granularity

message\_length=1+(nsymbols-1)\*nsamples+length(pulse)-1;

Nmin = max(message\_length,Nmin);

% %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

psd=zeros(Nfft,1);

for runs=1:nruns,

%random symbol sequence

symbols = sign(rand(nsymbols,1)-0.5);

nsymbols\_upsampled = 1+(nsymbols-1)\*nsamples;

symbols\_upsampled = zeros(nsymbols\_upsampled,1);

symbols\_upsampled(1:nsamples:nsymbols\_upsampled) = symbols;

message = conv(symbols\_upsampled,pulse);

%FM signal phase

% theta = 2\*pi\*kf\*ts\*cumsum(message);

% cenvelope = exp(j\*theta);

% time = (0:length(cenvelope)-1)\*ts;

% %freq domain signal computed using DFT

message\_freq = ts\*fft(message,Nfft); %FFT of size Nfft, automatically zeropads as needed

message\_freq\_centered = fftshift(message\_freq); %shifts DC to center of spectrum

psd=psd+abs(message\_freq\_centered).^2;

end

psd=psd/(nruns\*nsymbols);

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 PSD

plot(freqs,psd);

title(["PSD of message signal. kf = ", num2str(kf)]);

ylabel("Magnitude");

xlabel("Frequency (Hz)");

% xlim([-1.5, 1.5]);

print -dpng 8a.png

9.

oversampling\_factor = 16;

%for a pulse with amplitude one, the max frequency deviation is given by kf

kf=0.25;

%increase the oversampling factor if kf (and hence frequency deviation, and hence bw of FM s

oversampling\_factor = ceil(max(kf,1)\*oversampling\_factor);

ts=1/oversampling\_factor; %sampling time

nsamples = ceil(1/ts);

% pulse = ones(nsamples,1); %rectangular pulse

pulse\_time = 0:ts:1;

pulse = sin(pi\*pulse\_time);

% calculating PSD

nsymbols =1000;

symbols=zeros(nsymbols,1);

nruns=1000;

fs\_desired=0.1;

Nmin = ceil(1/(fs\_desired\*ts)); %minimum length DFT for desired frequency granularity

message\_length=1+(nsymbols-1)\*nsamples+length(pulse)-1;

Nmin = max(message\_length,Nmin);

% %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

psd=zeros(Nfft,1);

for runs=1:nruns,

%random symbol sequence

symbols = sign(rand(nsymbols,1)-0.5);

nsymbols\_upsampled = 1+(nsymbols-1)\*nsamples;

symbols\_upsampled = zeros(nsymbols\_upsampled,1);

symbols\_upsampled(1:nsamples:nsymbols\_upsampled) = symbols;

message = conv(symbols\_upsampled,pulse);

%FM signal phase

theta = 2\*pi\*kf\*ts\*cumsum(message);

cenvelope = exp(j\*theta);

time = (0:length(cenvelope)-1)\*ts;

% %freq domain signal computed using DFT

cenvelope\_freq = ts\*fft(cenvelope,Nfft); %FFT of size Nfft, automatically zeropads as needed

cenvelope\_freq\_centered = fftshift(cenvelope\_freq); %shifts DC to center of spectrum

psd=psd+abs(cenvelope\_freq\_centered).^2;

end

psd=psd/(nruns\*nsymbols);

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 PSD

plot(freqs,psd);

title(["PSD of fm signal. kf = ", num2str(kf)]);

ylabel("Magnitude");

xlabel("Frequency (Hz)");

xlim([-1.5, 1.5]);

print -dpng 9a.png

10.

oversampling\_factor = 16;

%for a pulse with amplitude one, the max frequency deviation is given by kf

kf=0.25;

%increase the oversampling factor if kf (and hence frequency deviation, and hence bw of FM s

oversampling\_factor = ceil(max(kf,1)\*oversampling\_factor);

ts=1/oversampling\_factor; %sampling time

nsamples = ceil(1/ts);

% pulse = ones(nsamples,1); %rectangular pulse

pulse\_time = 0:ts:1;

pulse = sin(pi\*pulse\_time);

% calculating PSD

nsymbols =1000;

symbols=zeros(nsymbols,1);

nruns=1000;

fs\_desired=0.1;

Nmin = ceil(1/(fs\_desired\*ts)); %minimum length DFT for desired frequency granularity

message\_length=1+(nsymbols-1)\*nsamples+length(pulse)-1;

Nmin = max(message\_length,Nmin);

% %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

psd=zeros(Nfft,1);

for runs=1:nruns,

%random symbol sequence

% symbols = sign(rand(nsymbols,1)-0.5);

symbols = randn(nsymbols,1);

nsymbols\_upsampled = 1+(nsymbols-1)\*nsamples;

symbols\_upsampled = zeros(nsymbols\_upsampled,1);

symbols\_upsampled(1:nsamples:nsymbols\_upsampled) = symbols;

message = conv(symbols\_upsampled,pulse);

%FM signal phase

theta = 2\*pi\*kf\*ts\*cumsum(message);

cenvelope = exp(j\*theta);

time = (0:length(cenvelope)-1)\*ts;

% %freq domain signal computed using DFT

cenvelope\_freq = ts\*fft(cenvelope,Nfft); %FFT of size Nfft, automatically zeropads as needed

cenvelope\_freq\_centered = fftshift(cenvelope\_freq); %shifts DC to center of spectrum

psd=psd+abs(cenvelope\_freq\_centered).^2;

end

psd=psd/(nruns\*nsymbols);

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 PSD

plot(freqs,psd);

title(["PSD of fm signal. kf = ", num2str(kf)]);

ylabel("Magnitude");

xlabel("Frequency (Hz)");

xlim([-1.5, 1.5]);

print -dpng 7a.png

12.

oversampling\_factor = 16;

%for a pulse with amplitude one, the max frequency deviation is given by kf

kf=0.25;

%increase the oversampling factor if kf (and hence frequency deviation, and hence bw of FM s

oversampling\_factor = ceil(max(kf,1)\*oversampling\_factor);

ts=1/oversampling\_factor;%sampling time

nsamples = ceil(1/ts);

pulse = ones(nsamples,1); %rectangular pulse

nsymbols =10;

symbols=zeros(nsymbols,1);

%random symbol sequence

% symbols = sign(rand(nsymbols,1)-0.5);

% symbols

symbols = [-1, 1, -1, 1, 1, -1, 1, 1, -1, -1]

%generate digitally modulated message

nsymbols\_upsampled=1+(nsymbols-1)\*nsamples;

symbols\_upsampled=zeros(nsymbols\_upsampled,1);

symbols\_upsampled(1:nsamples:nsymbols\_upsampled)=symbols;

message = conv(symbols\_upsampled,pulse);

%FM signal phase obtained by integrating the message

theta = 2\*pi\*kf\*ts\*cumsum(message);

cenvelope=exp(j\*theta);

L=length(cenvelope);

time=(0:L-1)\*ts;

Fc = 1000;

% cos(2pi\*Fc\*t + theta(t))

FM = cos(2\*pi\*Fc\*time.+theta');

subplot(2, 1, 1);

plot(time, message);

title(["Message signal. symbols = ", mat2str(symbols)]);

xlabel("time (s)");

ylabel("amplitude");

subplot(2, 1, 2);

plot(time, FM);

title("FM signal");

xlabel("time (s)");

ylabel("amplitude");

print -dpng 12a.png

13.

oversampling\_factor = 16;

%for a pulse with amplitude one, the max frequency deviation is given by kf

kf=4;

%increase the oversampling factor if kf (and hence frequency deviation, and hence bw of FM s

oversampling\_factor = ceil(max(kf,1)\*oversampling\_factor);

ts=1/oversampling\_factor;%sampling time

nsamples = ceil(1/ts);

pulse = ones(nsamples,1); %rectangular pulse

nsymbols =10;

symbols=zeros(nsymbols,1);

%random symbol sequence

% symbols = sign(rand(nsymbols,1)-0.5);

% symbols

symbols = [1, -1, -1, 1, -1, 1, 1, 1, -1, -1]

%generate digitally modulated message

nsymbols\_upsampled=1+(nsymbols-1)\*nsamples;

symbols\_upsampled=zeros(nsymbols\_upsampled,1);

symbols\_upsampled(1:nsamples:nsymbols\_upsampled)=symbols;

message = conv(symbols\_upsampled,pulse);

%FM signal phase obtained by integrating the message

theta = 2\*pi\*kf\*ts\*cumsum(message);

cenvelope=exp(j\*theta);

L=length(cenvelope);

time=(0:L-1)\*ts;

Fc = 1000;

% cos(2pi\*Fc\*t + theta(t))

FM = cos(2\*pi\*Fc\*time.+theta');

% -------- transmission ----------- %

% passing FM through differentiator

FM\_diff = [0;diff(FM')]/ts;

% diode filter - retaining only positive signal

FM\_diff\_diode = diodeFilter(FM\_diff');

% Envelope detector - RC filter

[time\_env, fm\_env] = RCfilter(time, FM\_diff\_diode, 0.04);

[time\_dcblock, fm\_dcblock] = DCblock(time\_env, fm\_env);

% fm\_dcblock = fm\_env;

% time\_dcblock = time\_env;

subplot(2, 1, 1);

plot(time, message);

title(["Message signal. symbols = ", mat2str(symbols)]);

xlabel("Time (s)");

ylabel("amplitude");

subplot(2, 1, 2);

plot(time\_dcblock, fm\_dcblock);

title("Demodulated signal");

xlim([0, 16]);

xlabel("Time (s)");

ylabel("amplitude");

print -dpng 13a.png

DCblock.m

%% DBblock: function description

function [time\_dcblock, signal\_dcblock] = DCblock(time, signal)

meanSignal = mean(signal)

time\_dcblock = time;

signal\_dcblock = signal.-meanSignal;

end

diodeFilter.m

%% diodeFilter: makes all values less than zero 0

function [result] = diodeFilter(vector)

vector(vector < 0) = 0;

result = vector;

end

RCfilter.m

%% RCFilter: function description

function [time\_f, signal\_f] = RCfilter(time, signal, RC = 0.383)

% t\_response = 0:ns/length(time):ns;

% 1/fc < RC < 1/b

% b = 1.5 KHz

t\_response = time;

dt = 1/40;

u\_response = ones(length(signal), 1);

% RC = 3.833 / 10;

% RC = 1 / 1.5;

temp\_t = t\_response./RC;

temp\_t = temp\_t.\*-1;

temp\_t\_exp = arrayfun( @(x) exp(x), temp\_t);

u\_response = u\_response.\*temp\_t\_exp;

u\_response = u\_response(1,:);

size(t\_response)

size(u\_response)

[time\_f, signal\_f] = contconv(signal, u\_response, time(1), t\_response(1), dt);

end