Software Lab 3.2: Frequency modulation basics

Lab Objectives: The goal of this lab to explore the characteristics of frequency modulated signals using digitally modulated messages, and to explore demodulation using differentiation in baseband.

Reading: Section 3.3 (angle modulation).

Laboratory Assignment

Consider the message signal $m(t) = \sum_n b[n]p(t - nT)$, where $b[n]$ are chosen from $\{-1, +1\}$, and T is the symbol interval. You can generate such a signal by modifying Code Fragment 2.3.2. Define a passband FM signal modulated by the message by

$$s_p(t) = \cos\left(2\pi f_c t + \theta(t)\right)$$

where f_c is the carrier frequency, and the phase

$$\theta(t) = 2\pi k_f \int_0^t m(\tau)d\tau$$

(assume $t \geq 0$). The complex envelope of s_p with respect to reference $2\pi f_c t$ is given by $s(t) = e^{j\theta(t)}$. We consider a digital message signal of the form $m(t) = \sum_n b[n]p(t - nT)$, where $b[n]$ are chosen independently and with equal probability from $\{-1, +1\}$.

The following code fragment generates the FM waveform for a rectangular pulse $I_{[0,1]}$.

```
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);
%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);
%plot I component
plot(time,Icomponent);
```

1) By modifying and enhancing the preceding code fragment as needed, plot the I and Q components of the complex envelope for a random sequence of bits as a function of time for $k_f = 0.25$. Also plot $\theta(t)/\pi$ versus t. How big are the changes in $\theta(t)$ corresponding to a given message bit $b[n]$? Do you notice a pattern in how the I and Q components depend on the message bits $\{b[n]\}$?
Remark: The special case of $k_f = 1/4$ is a digital modulation scheme known as Minimum Shift Keying (MSK). It can be viewed as FM modulation using a digital message, but the plots of the I and Q components should indicate that MSK can also be interpreted as the I and Q components each being amplitude modulated by a different set of bits, with an offset between the I and Q components.

2) Now redo (1) for $k_f = 4$. The patterns in the I and Q components are much harder to see now. We typically do not use such wideband FM for digital modulation, but may use it for analog messages.

3) For a complex baseband waveform $y(t) = y_c(t) + jy_s(t) = e(t)e^{j\theta(t)}$, we know that

$$\theta(t) = \tan^{-1} \frac{y_s(t)}{y_c(t)}$$

Show that

$$\frac{d}{dt}\theta(t) = \frac{y_c(t)y_s'(t) - y_s(t)y_c'(t)}{y_c^2(t) + y_s^2(t)} \tag{3.41}$$

For an FM signal, the message can be estimated as $\frac{1}{2\pi k_f}\frac{d}{dt}\theta(t)$, with the derivative computed using a highpass filter. Thus, this can be viewed as a baseband version of the limiter-discriminator demodulator for FM. It can be implemented using the following code fragment.

```
%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)./(I
```

4) Apply the preceding approach to the noiseless FM signals generated in parts 1) and 2). Plot the estimated message and the original message on the same plot, and comment on whether you are getting a good estimate.

5) Add an arbitrary phase to the complex envelope.

```
phi = 2*pi*rand; %phase uniform over [0,2 pi]
cenvelope = cenvelope*exp(j*phi);
%now apply baseband discriminator
```

Redo 4). What happens to the estimated message? Are you still getting a good estimate of the original message?

6) Now, add a frequency offset as well as a phase offset to the complex envelope.

```
phi = 2*pi*rand; %phase uniform over [0,2 pi]
df = 0.3;
cenvelope = cenvelope.*exp(j*(2*pi*df*time+phi));
%now apply baseband discriminator
```

Redo 4). What happens to the estimated message? Are you still getting a good estimate of the original message? If you are not quite getting the original message back, what can you do to fix the situation?

Remark: You should find that this crude differentiation technique does work for low noise (we are considering *zero* noise). However, it is rather fragile when noise is inserted. We do not explore this in this lab, but you are welcome to try adding Gaussian noise samples to the I and Q components and see how the discriminator performs for different values of noise variance. At the very least, one would need to lowpass filter the message estimate obtained above to average out noise, but it is far better to use feedback-based techniques such as the PLL for general FM demodulation, or, for digital messages, to use demodulation techniques that use the structure of the message. We do not discuss such techniques in this lab.

7) We now explore the spectral properties of FM. For the complex envelope $s(t) = e^{j\theta(t)}$, compute the Fourier transform numerically, choosing the length of the FFT (and hence n_s) so as to get a frequency resolution of 0.1. You can modify Code Fragment 2.5.1 or reuse code from Lab 1. Compute the power spectral density (PSD), defined as the magnitude squared of the Fourier transform divided by the interval over which you are computing it, and then averaged over multiple runs. Plot the PSD for $k_f = 1/4$ and $k_f = 4$. You can modify the following code fragment as needed.

```
nsymbols =1000;
symbols=zeros(nsymbols,1);
nruns=1000;
fs=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);
```

8) Plot the PSD for $k_f = \frac{1}{4}$ and $k_f = 4$. Are your results consistent with Carson's formula?

9) Redo 8), replacing the rectangular pulse by a sine pulse: $p(t) = \sin(\pi t)\, I_{[0,1]}(t)$. Are your results consistent with Carson's formula? Compare the spectrum occupancy in 8) and 9), commenting on the roles of k_f and $p(t)$.

```
pulse =transpose(sin(pi*(0:ts:1)));
```

10) Now, let us increase the dynamic range of the message by replacing the bits by numbers drawn from a Gaussian distribution with the same variance.

```
symbols = randn(nsymbols,1);
```

Compute and plot the PSD for a sinusoidal pulse, and compare with the spectral occupancy with that in 9).

11) Assuming that the unit of time is 1 ms, estimate the bandwidth of the FM signals whose PSDs you plotted in 9). You can either eyeball it, or estimate the length of the interval over which 95% of the signal power is contained.