Lab 9: \(M\)-ary Amplitude and Frequency Shift Keying, Signal Space

1 Introduction

Amplitude modulation (AM) can easily be used for the transmission of digital data if the (analog) message signal \(m(t)\) is replaced by a (digitally) pulse amplitude modulated (PAM) signal \(s(t)\). The simplest method to transmit binary data is on-off keying (OOK), whereby the transmitter is off for sending 0’s, and a carrier at frequency \(f_c\) is transmitted for sending 1’s (or vice versa). If the carrier oscillator at the transmitter is just turned on and off, then the phase for each 1 is random and a non-coherent receiver must be used. If phase coherence is maintained at the transmitter, then either coherent or non-coherent reception is possible. Instead of multiplying the carrier oscillator by 0 or 1 for binary data, it can be multiplied by \(-1\) or \(+1\), thereby producing phase changes by \(\pm 180^\circ\) at the carrier frequency \(f_c\). This method is called binary phase shift keying (BPSK) and requires a coherent receiver. An extension of this is phase shift keying (PSK) with four phases, e.g., using \(0^\circ, \pm 90^\circ\) and \(180^\circ\), for a 4-ary PAM signal. This is called quaternary phase shift keying (QPSK). Instead of using a single carrier frequency \(f_c\) for transmitting 0’s and 1’s, two distinct frequencies, e.g., \(f_{c0}\) and \(f_{c1}\) can be used to transmit 0’s and 1’s, respectively. The resulting signal is a binary frequency shift keying (BFSK) signal. It can be easily demodulated with a non-coherent receiver. If phase coherence is maintained at the transmitter, it can also be demodulated (with a smaller probability of error) using a coherent receiver. All three methods, amplitude shift keying (ASK), phase shift keying (PSK), and frequency shift keying (FSK) can be extended in a straightforward manner from the binary to the \(M\)-ary case by using \(M\) amplitudes, \(M\) phases, or \(M\) carrier frequencies, respectively.

1.1 Amplitude Shift Keying

The name “Amplitude Shift Keying” (ASK) refers to a digital bandpass modulation method, whereby the amplitude of a carrier frequency takes on different discrete values (at the sampling time instants), depending on the transmitted DT sequence \(a_n\). The blockdiagram of a coherent ASK communication system looks as follows.

![Block Diagram of ASK Communication System](image.png)
At the transmitter a PAM signal

\[ s(t) = \sum_{n=-\infty}^{\infty} a_n p(t - nT_B), \]

with baud rate \( F_B = 1/T_B \) and pulse \( p(t) \) is generated from the (digital) DT sequence \( a_n \). The CT signal \( s(t) \) then modulates a carrier with frequency \( f_c \) and phase \( \theta_c \), so that the transmitted signal is

\[ x(t) = A_c \sum_{n=-\infty}^{\infty} a_n p(t - nT_B) \cos(2\pi f_c t + \theta_c). \]

At the receiver the signal \( r(t) \) is demodulated using a local oscillator that replicates the carrier signal \( \cos(2\pi f_c t + \theta_c) \) with exact frequency and phase synchronization. After that, a regular baseband PAM receiver with matched filter (MF), matched to \( p(t) \ast h_{CL}(t) \), where \( h_{CL}(t) \) is the lowpass-equivalent channel response (i.e., \( h_C(t) \) shifted down in frequency by \( f_c \)), is used.

The simplest case of ASK is on-off keying (OOK), which is obtained when \( a_n \) is a unipolar binary sequence (i.e., \( a_n \in \{0, 1\} \)). The following two graphs show \( s(t) \) and \( x(t) \) for \( \{a_n\} = \{0, 1, 1, 1, 0, 1, 0, 1\} \), \( F_B = 100 \) Hz, rectangular \( p(t) \), \( f_c = 300 \) Hz, and \( \theta_c = -90^\circ \).

The PSD of a coherent OOK signal (i.e., an OOK signal for which \( \theta_c \) does not change over time) with rectangular \( p(t) \), \( f_c = 300 \) Hz, and \( F_B = 100 \) Hz is shown in the next graph.
If \( a_n \) is a polar binary sequence, e.g., \( \{a_n\} = \{-1, +1, +1, +1, -1, -1, +1, -1\} \), then \( s(t) \) and \( x(t) \) look as follows.

Assuming a rectangular \( p(t) \), the carrier is multiplied by either +1 or −1, depending on the data to be transmitted. This leads to phase changes by 180° and because of it this version of ASK is usually called binary phase shift keying (BPSK). Note that for BPSK it is crucial that the transmitter and receiver are phase-synchronized, i.e., BPSK requires a coherent receiver.

Rather than generating a baseband PAM signal first and then using amplitude modulation to make it a bandpass signal at \( f_c \), it is possible to directly generate a bandpass PAM signal...
using a pulse $p_x(t)$ that is already shifted to $f_c$ in the frequency domain. Mathematically, this can be expressed as

$$x(t) = \sum_{n=-\infty}^{\infty} a_n p_x(t - nT_B), \quad \text{where} \quad p_x(t) = p(t) \cos(2\pi f_c t + \theta_c).$$

At the receiver one can then use a MF that is directly matched to $p_x(t) * h_c(t)$ where $h_c(t)$ is the impulse response of the channel. The block diagram of an ASK transmission system that uses $p_x(t)$ directly at $f_c$ is shown below.

The simplest method to achieve wireless data transmission is to turn an oscillator with frequency $f_c$ on or off, depending on whether a one or a zero is transmitted. Using the morse code for on-off keying, this is how Marconi demonstrated in 1901 that transatlantic wireless communication was possible. What is likely to happen, however, if the oscillator at $f_c$ is turned on and off, is that the carrier phase $\theta_c$ takes on random values between 0 and $2\pi$. A model for this is shown in the following block diagram.

Here it is assumed that for each new symbol $a_n$ a random carrier phase value $\theta_c[n]$ is chosen (even if the oscillator is not turned off between two consecutive ones). Thus

$$x(t) = A_c \sum_{n=-\infty}^{\infty} a_n p(t - nT_B) \cos(2\pi f_c t + \theta_c[n])$$

$$= A_c \left[ \sum_{n=-\infty}^{\infty} a_n \cos \theta_c[n] p(t-nT_B) \cos 2\pi f_c t - \sum_{n=-\infty}^{\infty} a_n \sin \theta_c[n] p(t-nT_B) \sin 2\pi f_c t \right],$$

where $\theta_c[n] \in [0, 2\pi)$. Because the transmitted carrier phase of such a signal is not constant, the resulting ASK signal will be called **non-coherent ASK**. An example when $\{a_n\} = \{0, 1, 1, 1, 0, 0, 1, 0\}$, $F_B = 100$ baud, and $f_c = 300$ Hz is shown in the next graph.
To demodulate a signal with (unknown) random phase, a **matched filter envelope detector (MFED)**, such as the one shown in the following block diagram, can be used.

In the process of computing $b(t) = \sqrt{w_i^2(t) + w_q^2(t)}$, the dependence on the carrier phase (and in fact the dependence on the exact carrier frequency $f_c$) drops out. The interesting observation to be made is that, because of the orthogonality of $\cos 2\pi f_c t$ and $\sin 2\pi f_c t$, the in-phase component $w_i(t)$ and the quadrature component $w_q(t)$ span a 2-dimensional space. Upon sampling at times $t = nT_B$, as shown in the block diagram below, this becomes a **2-dimensional signal space** spanned by $w_i[n]$ (real part in complex plane) and $w_q[n]$ (imaginary part in complex plane).
Thus, coherent ASK uses a one-dimensional signal space and noncoherent ASK uses a two-dimensional signal space. This is shown for OOK in the following figure which is called a scatter plot.

Note that for noncoherent ASK the signal “points” are actually points on concentric circles. If the received signal is noisy, then the noise manifests itself as deviations of these points from the concentric circles, as shown for OOK with SNR $E_b/N_0 = 12$ dB in the scatter plot below.
The black stars are the signal points of the noncoherent ASK signal at the transmitter. The blue and green circles are the received noisy signal points with blue indicating a decision for a received 0 and green indicating a decision for a received 1. As the SNR decreases, the received signal points deviate more and more from their nominal positions at the origin or on the circle with radius 1, and the probability of symbol error, \( P_s(\mathcal{E}) \), approaches 0.5.

### 1.2 Frequency Shift Keying

Rather than using a digital DT sequence to change the amplitude of a carrier, the frequency of the carrier can be changed by the data sequence to be transmitted. The resulting digital bandpass signaling method is called **frequency shift keying (FSK)**. One way to implement \( M \)-ary FSK is shown in the following block diagram.
An $M$-ary DT sequence $d_n$, with $d_n \in \{0, 1, \ldots, M-1\}$, is first demultiplexed into $M$ binary sequences, the first one having 1’s in all positions where $d_n$ is zero and 0’s everywhere else, the second one having 1’s in all positions where $d_n$ is one and 0’s everywhere else, etc. The $M$ sequences are then converted to OOK signals of the form

$$x_m(t) = \sum_{n=-\infty}^{\infty} \delta[d_n - m] p_m(t - nT_B), \quad m = 0, 1, \ldots, M-1,$$

and added up to produce the FSK signal $x(t)$ as

$$x(t) = \sum_{m=0}^{M-1} x_m(t) = \sum_{m=0}^{M-1} \sum_{n=-\infty}^{\infty} \delta[d_n - m] p_m(t - nT_B).$$

Depending on whether the individual OOK signals are coherent or non-coherent, one can distinguish between **coherent FSK** and **non-coherent FSK**. The blockdiagrams below show how to implement the individual bandpass PAM functions $p_m(t)$, $m = 0, 1, \ldots, M-1$, in each case.

Thus, for coherent FSK $p_m(t) = A_c p(t) \cos(2\pi f_{cm} t + \theta_{cm})$, and for non-coherent FSK $p_m(t) = A_c p(t) \cos(2\pi f_{cm} t + \theta_{cm}[n])$. Note that, when the FSK signal is made up from the outputs of individual oscillators, $x(t)$ may contain phase jumps at the symbol boundaries, as can be seen in the following graph that shows an example of coherent binary FSK with rectangular $p(t)$, $F_B = 100$ baud, $f_{c1} = 300$ Hz, $\theta_{c1} = -90^\circ$, $f_{c2} = 400$ Hz, $\theta_{c2} = -90^\circ$. 

![Diagram showing implementation of coherent and non-coherent FSK with PAM functions](image-url)
The upper graph shows \( x(t) \) in the time domain for \( \{d_n\} = \{0, 1, 1, 0, 1, 0\} \), and the lower graph shows the PSD \( S_x(f) \) when \( d_n \) is a random binary signal with equally likely 0’s and 1’s. The next graph shows \( x(t) \) for the same data and parameters, but using a non-coherent FSK transmitter.

The block diagram below shows the basic structure of a receiver for \( M \)-ary FSK.
There is a matched filter at each of the $M$ FSK frequencies $f_{c0}, \ldots, f_{cM-1}$. Coherent FSK can be received with coherent matched filters. One way to implement a coherent MF when $f_{cm}$ and $\theta_{cm}$ are fixed and known is shown in the following blockdiagram.

An interesting question is how close together any two adjacent frequencies, say $f_{c\ell}$ and $f_{cm} = f_{c\ell} + \Delta f$, should be chosen. By making $\Delta f$ small, the total bandwidth used by $M$-ary FSK can be reduced, but if $\Delta f$ is too small, then the normalized distance between symbols using different frequencies becomes too small and thus the probability of error increases. As a general criterion, one usually tries to make all $M$ symbols in $M$-ary FSK orthogonal, i.e., assuming $p(t)$ is a rectangular pulse of width $T_B$, one selects $\Delta f$ in $f_{cm} = f_{c\ell} + \Delta f$ such that

$$\int_{(n-1/2)T_B}^{(n+1/2)T_B} \cos(2\pi f_{c\ell}t + \theta_{c\ell}) \cos(2\pi f_{cm}t + \theta_{cm}) \, dt = 0 , \quad \text{for all integers } n.$$

The scatter plot in the following figure shows $w_0[n]$ versus $w_1[n]$ for coherent binary FSK satisfying the orthogonality condition.
For the graph above a FSK signal with high SNR (40 dB) was used so that the nominal signal points (black *) and the received signal points (red o) coincide. The graph below shows how this changes when the SNR is reduced to $E_b/N_0 = 12$ dB.
If the carrier phase is random and/or unknown to the receiver, then matched filter envelope detectors (MFED) of the form shown in the blockdiagram below need to be used.

\[
2 \cos 2\pi f_{cm}t \times \sqrt{w_{mi}(t)^2 + w_{mq}(t)^2} \times v_{mi}(t) \xrightarrow{\text{MF}} w_{mi}(t) \quad \text{and} \quad 2 \cos 2\pi f_{cm}t \times \sqrt{w_{mi}(t)^2 + w_{mq}(t)^2} \times v_{mq}(t) \xrightarrow{\text{MF}} w_{mq}(t)
\]

In this case, a scatter plot of binary FSK would have to be drawn in 4-dimensional space. Looking at just \(w_{iq}[n]\) versus \(w_{iq}[n]\), or \(w_{iq}[n]\) versus \(w_{iq}[n]\) yields signal “points” that either lie at the origin or on a circle, centered at the origin.

One more version of FSK that is actually preferred in practice is **continuous phase FSK (CPFSK)**. This can be obtained from coherent FSK by choosing the frequencies \(f_{cm}\) and the phases \(\theta_{cm}\) in such a way that phase jumps are avoided at the symbol boundaries. An example of binary CPFSK and its PSD is shown in the following two graphs.
As can be seen in the lower graph, the absence of phase jumps in \( x(t) \) yields a PSD \( S_x(f) \) that decreases rapidly for frequencies below \( f_{c0} \) and above \( f_{cM-1} \) (in the \( M \)-ary case), which reduces interference between communication systems that use adjacent bands.

1.3 Signal Space, Probability of Error

The graph below shows the probability \( P_s(\mathcal{E}) \) of error for OOK, BPSK and BFSK when coherent receivers are used.

In general a non-coherent receiver will have a larger \( P_s(\mathcal{E}) \) for a given \( E_b/N_0 \). The reason for this is that a coherent OOK/ASK receiver at some carrier frequency \( f_c \) (or \( f_{ci} \) for the \( i \)-th FSK frequency) only needs to look at the in-phase component of the demodulation, whereas a non-coherent receiver needs to look at both the in-phase and the quadrature components. Thus, a coherent receiver only picks up noise from the in-phase channel, whereas a non-coherent receiver picks up noise from both the in-phase and the quadrature channels.

1.4 Digital Modulation in GNU Radio

The GNU Radio Companion (version 3.7.6.1) comes with a number of specific digital modulator blocks, e.g., for CPFSK (continuous phase FSK), for DPSK (differential PSK), GFSK (Gaussian FSK), and GMSK (Gaussian minimum shift keying). There is also a general
Constellation Modulator whose properties can be controlled using a Constellation Object block. This modulator offers more possibilities to experiment with different parameters and constellations and we can use it to generate OOK, BPSK, and BFSK signals.

The figure below shows a baseband modulator for OOK.

The Random Source block generates bytes with random values in the range from 0 to 255 which are fed to the input of the Constellation Modulator via a Throttle block. With the settings chosen in the above block diagram 100 random bytes are generated and then repeated. The basic sample rate is set to 100k samples per second and the Throttle block limits the dataflow to (approximately) 100,000 bits per second. The basic features of the modulator are set to not use differential encoding, to use 8 samples per symbol via the sps variable (thus the baud rate is one eighth of the sample rate), and an excess bandwidth of 0.35 (i.e., $\alpha = 0.35$ for the root raised cosine pulse shape). The data input to the modulator is in the form of (8-bit) bytes which are transmitted serially, MSB first. The signal points and their binary representation are set by the Constellation Object. Here we use symbols 0 and 1 which are mapped to constellation points 0 (“off”) and 1 (“on”), respectively. Because the Constellation Modulator assumes a distance of 2 between the two constellation points, the (generally complex-valued) output is multiplied by 0.5 to obtain a signal amplitude of 1. The output of the modulator is displayed using a QT GUI Sink and it is also recorded in a file for further processing, e.g., in Matlab. The PSD of the generated OOK signal has the following shape.
The dc component that results from OOK is clearly visible. Not also the shape and the bandwidth of the spectrum. The pulse amplitude modulator (PAM) uses a rrc (root raised cosine) or rrcf (root raised cosine in frequency) pulse shape with an excess bandwidth of $\alpha = 0.35$ in the example shown. Since 8 samples per symbol and a (bit) sample rate of 100 kHz were used, the overall (-6 dB) bandwidth of the signal is $BW = 1.35 \times 10^5 / (2 \times 8) = 8437.5$ kHz. In the time domain the signal is purely real (because of the way the constellation was specified) and the bit interval is $8 \times 10^{-5} = 80 \mu s$, as shown in the next graph.

To obtain the baseband constellation configuration, the real part of the baseband signal is plotted along the horizontal axis and the imaginary part is plotted along the vertical axis at the optimum sampling time. Since 8 samples per symbol are used in these examples, a decimating low-pass filter with a decimation factor of 8 was used before displaying the signal in a QT GUI Constellation Sink. The result is shown next.
The expected result is two points, one at the origin (when the OOK signal is off) and one at (1,0) (when the OOK signal is on). Instead, two line segments are displayed. The two main reasons for this are that the Constellation Sink is not exactly synchronized with the optimum sampling time, and that the rrc pulse creates intersymbol interference during transmission (which is removed when the receiver uses a filter matched to the rrc pulse). To analyze the generated lowpass OOK signal in more detail we recorded the signal in GNU Radio to a file (digMod_001.bin) and imported it in Matlab using the following commands:

```matlab
Fs2 = 100000; % Sampling rate
tlen = 5; % Length of signal in sec
N = 2*tlen*Fs2; % Number of samples to read
fID = fopen('digMod_001.bin','r');,sps = 8;
rbin = fread(fID,N,'float'); % Read N 32-bit floating point numbers
fclose(fID);
FB = Fs2/sps;
rt = rbin(1:2:end) + j*rbin(2:2:end);
rt = rt.'; % Convert to complex-valued row vector
tt = [0:length(rt)-1]/Fs2; % Generate time axis
```

Note that complex numbers are recorded in the GNU Radio file as pairs of 32-bit (IEEE standard) floating point numbers (real part followed by imaginary part). Thus, the file size of 5 seconds of complex-valued data recorded with a sample rate of 1 MHz is $5 \times 8 \times 10^6 = 40$ MBytes. To convert the data to complex numbers in Matlab add the odd indexed floating point numbers (Matlab starts with index 1) to $j$ times the even indexed numbers. The graph below shows the PSD of the OOK signal in Matlab.
Below are the eye diagrams of the OOK signal before and after the matched filter.

The figure below shows the signal $b(t)$ after the matched filter (blue line) and the samples at $kT_b$ (red circles).
The autocorrelation function of the demodulated digital data shows the period of 800 which results from the repetition of 100 random bytes.

The blockdiagram below implements the bandpass version of a OOK modulator.
The baseband OOK signal is upsampled to a sampling frequency of 1 MHz, lowpass filtered to 40 kHz and then multiplied by a (complex-valued) sinusoidal carrier with a frequency of 100 kHz. Now the time domain display has the characteristic appearance of a bandpass OOK signal and looks like this.

![Image of time domain display](image)

A binary frequency-shift keying (BFSK) signal can be generated by using two (orthogonal) OOK signals. In the following block diagram this is achieved by using two modulators with complementary symbol mappings (0,1 and 1,0) and orthogonal constellation points (0,1 and 0,1j).

![Image of block diagram](image)

The spectrum of the resulting (complex-valued) lowpass version of the BFSK signals is shown next. Note again the dc component that results from the two OOK signals.
The time domain plot below shows the complementary nature of the two signals that make up the BFSK signal.

The constellation plot shows that the two signal “points” (line segments due to imperfect sample timing and due to intersymbol interference) lie on orthogonal axes.
To obtain a (true) bandpass BPSK signal, the outputs of the two modulators are multiplied by two different carrier frequencies (typically spaced a multiple of the baud rate $F_B$ apart). Note that the constellation points for both modulators were changed back to 0,1 because the orthogonality between the two signal points is now achieved by using carrier frequencies that are orthogonal over a symbol time $T_B = 1/F_B$. The resulting block diagram is shown below.

Using two carrier frequencies that are sufficiently different the nature of the (complex-valued) bandpass BFSK signal can easily be visualized as seen in the plot below.
By a simple change of the constellation points for OOK from 0,1 to -1,1 a binary phase-shift keying (BPSK) signal is obtained as shown in the next block diagram.

Now the spectral line at dc is gone because the dc-component of a BPSK signal is zero if 0’s and 1’s are equally likely.
In the time domain plot the signal varies in amplitude between -1 and +1 (with some overshoot due to the rrc pulse shaping).

The signal constellation for BPSK ideally has two points along the horizontal axis, one at -1 and one at +1. Due to non-perfect sampling synchronization and intersymbol interference this manifest itself again as two line segments along the horizontal axis as seen in the figure below.
Finally, the lowpass BPSK signal gets upsampled and multiplied by a sinusoidal carrier signal to become a (complex-valued) bandpass BPSK signal. The blockdiagram for this is shown next.

In the time domain we now have an RF signal with the modulated digital data embedded in the form of $180^\circ$ phase changes.

Using Volk machine: avx_64_mmx_orc
>>> Done (return code:9)
Of the three modulation methods, OOK, BFSK, and BPSK discussed here, BPSK performs best for a given signal-to-noise ratio. This comes at the expense of the need for a significantly more complex coherent receiver.

2 Lab Experiments

E1. Amplitude Shift Keying. (a) Use your pam12 function as a building block to complete the ASK transmitter function askxmtr whose header is shown below.
function [xt,tt,st] = askxmtr(anthcn,FB,Fs,ptype,pparms,xtype,fcparms)
%askxmtr Amplitude Shift Keying (ASK) Transmitter for
% Choherent ('coh') and Non-coherent ('noncoh') ASK Signals
% >>>>> [xt,tt,st] = askxmtr(anthcn,FB,Fs,ptype,pparms,xtype,fcparms) <<<<<
% where
% xt: transmitted ASK signal, sampling rate Fs
% x(t) = s(t)*cos(2*pi*fc*t+(pi/180)*thetac)
% tt: time axis for x(t), starts at t=-TB/2
% st: baseband PAM signal s(t)
% xtype: Transmitter type from list {'coh','noncoh'}
% anthcn = [an] for {'coh'}
% anthcn = [an;thetacn] for {'noncoh'}
% an: N-symbol DT input sequence a_n, 0<=n<N
% thetacn: N-symbol DT sequence theta_c[n] in degrees,
% used instead of thetac for {'noncoh'} ASK
% FB: baud rate of a_n (and theta_c[n]), TB=1/FB
% Fs: sampling rate of x(t), s(t)
% ptype: pulse type from list
% {'man','rcf','rect','rrcf','sinc','tri'}
% pparms = not used for {'man','rect','tri'}
% pparms = [k alpha] for {'rcf','rrcf'}
% pparms = [k beta] for {'sinc'}
% k: "tail" truncation parameter for {'rcf','rrcf','sinc'}
% (truncates at -k*TB and k*TB)
% alpha: Rolloff parameter for {'rcf','rrcf'}, 0<=alpha<=1
% beta: Kaiser window parameter for {'sinc'}
% fcparms = [fc thetac] for {'coh'}
% fcparms = [fc] for {'noncoh'}
% fc: carrier frequency
% thetac: carrier phase in deg (0: cos, -90: sin)

To generate noncoherent ASK signals with a random carrier sequence $\theta_c[n]$, use in-phase and quadrature PAM signals $s_i(t)$ and $s_q(t)$ and QAM modulation as shown in the following blockdiagram.
Test your transmitter by generating a short (about 10 symbol times) random coherent OOK signal and a short random noncoherent OOK signal and displaying them in the time domain. Use $F_s = 44100$ Hz, $F_B = 100$ baud, $f_c = 300$ Hz, and a rectangular pulse $p(t)$ of width $T_B = 1/F_B$. Generate a uniformly distributed random phase for noncoherent OOK using the `rand` function.

(b) Complete the following ASK receiver function, called `askrcvr`. The goal is to be able to use it to receive coherent and noncoherent ASK signals, and to make scatter plots.
function [bn,win,wqn,ixn]=askrcvr(tt,rt,rtype,fcparms,FBparms,ptype,pparms)
%askrcvr Amplitude Shift Keying (ASK) Receiver for
% Coherent ('coh') and Non-coherent ('noncoh') Reception
% >>>>> [bn,win,wqn,ixn] =
% = askrcvr(tt,rt,rtype,fcparms,FBparms,ptype,pparms) <<<<<
% where
% bn: received DT sequence
% win: in-phase component of bn
% wqn: quadrature component of bn
% ixn: sampling indexes for b(t), w(t) to obtain bn, wn
% tt: time axis for r(t)
% rt: received (noisy) ASK signal r(t)
% rtype: Receiver type from list {'coh','noncoh'}
% fcparms = [fc thetac] for {'coh'}
% fcparms = [fc] for {'noncoh'}
% FBparms = [FB dly]
% FB: baud rate of PAM signal, TB=1/FB
% dly: sampling delay for b(t) -> b_n, fraction of TB
% Sampling times are t=n*TB+t0 where t0=dly*TB
% ptype: pulse type from list
% {'man','rcf','rect','rrcf','sinc','tri'}
% pparms = not used for {'man','rect','tri'}
% pparms = [k alpha] for {'rcf','rrcf'}
% pparms = [k beta] for {'sinc'}
% k: "tail" truncation parameter for {'rcf','rrcf','sinc'}
% (truncates at -k*TB and k*TB)
% alpha: rolloff parameter for {'rcf','rrcf'}, 0<=alpha<=1
% beta: kaiser window parameter for {'sinc'}

Test askrcvr together with askxmtr using random (unipolar) binary data and the parameters given in part (a).

(c) Let $F_s = 44100$ Hz, $F_B = 100$ baud, $f_c = 2100$ Hz, and let $p(t)$ be a rectangular pulse of width $T_B$. Use random binary data to produce (i) a coherent OOK signal, (ii) a noncoherent OOK signal, and (iii) a BPSK signal, each of duration 2 sec. Plot and compare the PSDs for all three cases. Then use the win and wqn outputs of the askrcvr function to make scatter plots for the three signals and compare them. Are there any interesting spectral lines if you look at the PSDs of the squared ASK signals? How do things change if you use a triangular pulse $p(t)$ (of total width $2T_B$ from $-T_B$ to $+T_B$) instead of the rectangular pulse?

(d) The binary ASK signals in the wav files asksig901.wav and asksig902.wav contain 8-bit, LSB-first, ASCII signals. Analyze the two signals and extract the text messages. **Hint:** The signals in the wav files have been scaled so that their amplitude (including noise) is less than or equal to one. Use eye diagrams to determine the proper threshold below which the receiver decides that a 0 was received and above which it decides that a 1 was received.
E2. Frequency Shift Keying. (a) Write a Matlab function called `fskxmtr` which implements an $M$-ary FSK transmitter for either coherent or noncoherent FSK. The header of this function is shown below.

```matlab
function [xt,tt] = fskxmtr(M,dnthcn,FB,Fs,ptype,pparms,xtype,fcparms)
%fskxmtr $M$-ary Frequency Shift Keying (FSK) Transmitter for
% Coherent ('coh') and Non-Coherent ('noncoh') FSK Signals
% >>>>> [xt,tt] = fskxmtr(M,dnthcn,FB,Fs,ptype,pparms,xtype,fcparms) <<<<<
% where
% xt: transmitted FSK signal, sampling rate Fs
% tt: time axis for x(t), starts at t=-TB/2
% M: number of distinct symbol values in d_n
% xtype: Transmitter type from list {'coh','noncoh'}
% dnthcn = [dn] for {'coh'}
% dnthcn = [dn;thetacn] for {'noncoh'}
% dn: $M$-ary (0,1,...,M-1) N-symbol DT input sequence d_n
% thetacn: N-symbol DT sequence theta_c[n] in degrees,
% used instead of thetac0..thetacM-1 for {'noncoh'} FSK
% FB: baud rate of d_n (and theta_c[n]), TB=1/FB
% Fs: sampling rate of x(t)
% ptype: pulse type from list
% {'man','rcf','rect','rrcf','sinc','tri'}
% pparms = [] for {'man','rect','tri'}
% pparms = [k alpha] for {'rcf','rrcf'}
% pparms = [k beta] for {'sinc'}
% k: "tail" truncation parameter for {'rcf','rrcf','sinc'}
% (truncates at -k*TB and k*TB)
% alpha: Rolloff parameter for {'rcf','rrcf'}, 0<=alpha<=1
% beta: Kaiser window parameter for {'sinc'}
% fcparms = [fc0 fc1 .. fcM-1;thetac0 thetac1 .. thetacM-1] for {'coh'}
% fcparms = [fc0 fc1 .. fcM-1] for {'noncoh'}
% fc0 fc1 .. fcM-1: FSK (carrier) frequencies
% (0: cos, -90: sin) for {'coh'}
% thetac0 thetac1 .. thetacM-1: FSK (carrier) phases in deg
```

Test `fskxmtr` by recreating the three (time domain) sample graphs for $F_B = 100$ baud, $f_{c0} = 300$ Hz and $f_{c1} = 400$ Hz, which were given in the introduction for binary coherent FSK, noncoherent FSK, and CPFSK. Use $d_n = \{0,1,1,0,0,1,0\}$ and (for the second graph) $\theta_c[n] = \{270^\circ, 225^\circ, 4^\circ, 135^\circ, 250^\circ, 90^\circ, 40^\circ, 240^\circ\}$.

(b) Implement the FSK receiver function `fskrsvr` whose header is given below.
function [bn,win,wqn,ixn]=fskrcvr(M,tt,rt,rtyp,fcparms,FBparms,ptype,pparms)

% fskrcvr M-ary Frequency Shift Keying (FSK) Receiver for
% Coherent ('coh') and Non-coherent ('noncoh') FSK Reception
% >>>>> [bn,win,wqn,ixn] =
% = fskrcvr(M,tt,rt,rtyp,fc parms,FBparms,ptype,pparms) <<<<<
% where
% bn: received DT sequence
% win = [w0in;w1in;...;wM-1in] in-phase matched filter outputs
% for {'coh','noncoh'}
% wqn = [w0qn;w1qn;...;wM-1qn] quadrature matched filter outputs
% for {'coh','noncoh'}
% ixn: sampling indexes for b(t), w(t) to obtain bn, wn
% M: number of distinct FSK frequencies
% tt: time axis for r(t)
% rt: received (noisy) FSK signal r(t)
% rtyp: Receiver type from list {'coh','noncoh'}
% fcparms = [fc0 fc1 .. fcM-1;thetac0 thetac1 .. thetacM-1]
% for {'coh'}
% fcparms = [fc0 fc1 .. fcM-1] for {'noncoh'}
% fc0 fc1 .. fcM-1: FSK (carrier) frequencies
% for {'coh','noncoh'}
% thetac0 thetac1 .. thetacM-1: FSK (carrier) phases in deg
% (0: cos, -90: sin) for {'coh'}
% FBparms = [FB dly]
% FB: baud rate of PAM signal, TB=1/FB
% dly: sampling delay for b(t) -> b_n, fraction of TB
% Sampling times are t=n*TB+t0 where t0=dly*TB
% ptype: pulse type from list {'man','rcf','rect','rrcf','sinc','tri'}
% pparms = [] for {'man','rect','tri'}
% pparms = [k alpha] for {'rcf','rrcf'}
% pparms = [k beta] for {'sinc'}
% k: "tail" truncation parameter for {'rcf','rrcf','sinc'}
% (truncates at -k*TB and k*TB)
% alpha: rolloff parameter for {'rcf','rrcf'}, 0<=alpha<=1
% beta: Kaiser window parameter for {'sinc'}

Depending on the choice of the rtyp, the receiver should perform demodulation using either M coherent MFs or M MFEDs. Test both receiver modes with the signals that you generated in (a) when you were testing fskxmtr.

(c) Generate a coherent binary FSK signal from random data with equally likely 0’s and 1’s, using a rectangular p(t), \( F_B = 100 \) baud, \( f_{c0} = 300 \) Hz, \( \theta_{c0} = 0^\circ \), \( f_{c1} = 400 \) Hz, and \( \theta_{c1} = 0^\circ \). Use a coherent demodulator to produce a scatter plot of \( w_0[n] \) versus \( w_1[n] \). Change the phase \( \theta_{c1} \) from 0° to 180° and check whether the signals transmitted at \( f_{c0} \) and at \( f_{c1} \) remain
orthogonal in the signal space spanned by $w_0[n]$ and $w_1[n]$. Is it possible to reduce the frequency spacing $\Delta f = f_{c1} - f_{c0}$ to a value less than $F_B$ while maintaining orthogonality? Try changing $f_{c1}$ to 350 Hz and vary the phase $\theta_{c1}$ again from 0° to 180°.

(d) Use uniformly distributed random $M$-ary data of length about 2 sec to generate PSD plots of $M = 2$ and $M = 4$ coherent, noncoherent, and continous-phase FSK (CPFSK) signals with rectangular $p(t)$. Determine the -40 dB bandwidth in all cases. Use $F_B = 100$ baud, $f_{c0} = 2100$ Hz, and $f_{cm} = f_{c0} + mF_B$. For coherent FSK set $\theta_{cm} = 0$ for all $m$. For CPFSK choose the phases $\theta_{cm}$ such that there are no phase jumps.

(e) The wav files `fsksig901.wav` and `fsksig902.wav` contain binary FSK signals made from 8-bit, LSB-first, ASCII encoded characters. Analyze the two signals and extract the text messages.

(f) The binary GNU Radio file `digMod_905.bin` contains several (complex-valued) binary communication signals. Use the GNU Radio Companion to find the signals recorded in the file and determine their properties such as the type of modulation and the carrier frequencies. Each of the signals contains an ASCII coded (MSB first) message. Try to demodulate the signals and extract the messages.

E3. $P_s(\xi)$ for Coherent/Noncoherent ASK and FSK. (Experiment for ECEN 5002, optional for ECEN 4652) (a) Generate a DT sequence $a_n$ consisting of $N = 10000$ random binary symbols with equally likely values of 0 and 1. Use $a_n$ to generate a coherent OOK signal $x(t)$ with rectangular $p(t)$ of width $T_B$, $F_B = 1/T_B = 100$ baud, $f_c = 2100$ Hz, and $\theta_c = -90^\circ$. Add white Gaussian noise to $x(t)$ to create a received signal $r(t)$ with SNR $E_b/N_0$ in the range 4 . . . 12 dB. Demodulate $r(t)$ with a coherent matched filter receiver and compare the received sequence (after proper quantization) with the transmitted sequence. Compute the probability of symbol error $P_s(\xi)$ and verify that you obtain results that are very similar to the ones shown in the $P_s(\xi)$ graph given in the introduction. Then use the same transmitted signal, but a noncoherent receiver and determine the resulting (approximate) loss in $E_b/N_0$ (in dB) compared to coherent reception.

(b) Repeat (a) for a coherent binary FSK signal using a rectangular $p(t)$, $F_B = 100$ baud, $f_{c0} = 2100$ Hz, $\theta_{c0} = 0^\circ$, $f_{c1} = 2200$ Hz, and $\theta_{c1} = 180^\circ$. Determine the (approximate) loss in $E_b/N_0$ (in dB) compared to coherent reception when a noncoherent MFED receiver is used.