AMPLITUDE ESTIMATION OF MINIMUM SHIFT
KEYING IN CO-CHANNEL INTERFERENCE

A thesis presented to
the faculty of
the Russ College of Engineering and Technology of
Ohio University

In partial fulfillment
of the requirements for the degree
Master of Science

Ravikanth Ekanthalingam
November 2004
This thesis entitled

AMPLITUDE ESTIMATION OF MINIMUM SHIFT KEYING
IN CO-CHANNEL INTERFERENCE

BY
RAVIKANTH EKANTHALINGAM

has been approved for
the School of Electrical Engineering and Computer Science
and the Russ College of Engineering and Technology by

David Matolak
Assistant Professor of Electrical Engineering and Computer Science

Dennis Irwin
Dean, Russ College of Engineering and Technology
Amplitude Estimation of Minimum Shift Keying in the presence of Co-channel interference (130pp.)

Director of Thesis: David W. Matolak

We investigate the estimation of amplitude of minimum shift keying in the presence of co-channel interference. We consider the maximum likelihood (ML) estimation technique for amplitude estimation, for the practical case of sampled waveforms. We assume a single interfering user, and derive the full ML estimator as a function of the number of samples per symbol, and the sampling phase. We review analysis for corresponding rectangular pulse cases for insight, and also consider the effect of both bit and carrier phase synchronism on performance. Using high signal-to-noise ratio approximations, we show that the ML estimator for MSK becomes a scaled version of that used for rectangular pulses, with the scaling dependent upon the sampling rate and sampling phase. The accuracy of the amplitude estimator was verified by performing a single stage of interference cancellation on a practical system consisting of two users in the presence of Additive White Gaussian Noise. Numerical results are included to corroborate the analysis.

Approved:_______________________________________________________________

David W. Matolak

Assistant professor,

School of Electrical Engineering and Computer Science
Acknowledgements

I would like to express my sincere appreciation and gratitude to my advisor Dr. David Matolak. He has been a constant source of inspiration and support to me. This thesis would have not been possible without his guidance. I feel honored and fortunate to have been associated with him. I would like to express my appreciation to my committee members Dr. Jeffrey Dill, Dr. Chris Bartone and Dr. Sergio Lopez who have been very helpful.

Finally, most of all, I would like to thank my parents and my brother for their love and affection.
# TABLE OF CONTENTS

Abstract........................................................................................................................................... 3  
Acknowledgements......................................................................................................................... 4  
List of Figures ................................................................................................................................. 7  

## CHAPTER 1 ................................................................................................................................. 9  
INTRODUCTION............................................................................................................................. 9  
1.1 Background-Basic Communication System.............................................................................. 9  
1.2 Applications of MSK-GSM & IC .......................................................................................... 10  
1.3 Interference Cancellation ......................................................................................................... 12  
1.4 Thesis Scope ............................................................................................................................ 14  

## CHAPTER 2 .............................................................................................................................. 15  
SIGNAL AND SYSTEMS MODELS .......................................................................................... 15  
2.1 Phase Modulations ................................................................................................................... 15  
  2.1.1 QPSK Signaling ................................................................................................................. 16  
  2.1.2 OQPSK Signaling ............................................................................................................. 19  
  2.1.3 MSK Signaling ................................................................................................................... 21  
  2.1.4 Power Spectra ................................................................................................................... 25  
2.2 Problem Setting ....................................................................................................................... 27  
2.3 System Model ......................................................................................................................... 29  
  2.3.1 Interference Cancellation ................................................................................................. 32  

## CHAPTER 3 .............................................................................................................................. 34  
AMPLITUDE ESTIMATION ......................................................................................................... 34  
3.1 Assumptions-Frequency coherence ......................................................................................... 34  
3.2 Maximum Likelihood Estimate (MLE) .................................................................................. 46  
  3.2.1 Case 1: MLE for Rectangular pulses in AWGN ................................................................. 47  
  3.2.2 Normalized Bias of the Amplitude Estimate ................................................................. 52
3.2.3 Case 2: MLE for Sinusoidal Pulses, Bit Synchronous Case..........................55
3.2.4 Case 3: Rectangular Pulses, Two Users in AWGN.....................................58
3.2.5 Case 4: Sinusoidal Pulses, Two Users in AWGN.......................................61

CHAPTER 4.............................................................................................................63
NUMERICAL RESULTS..........................................................................................63
4.1 Single User in AWGN.....................................................................................64
4.2 Two Users in AWGN......................................................................................68

CHAPTER 5.............................................................................................................74
SUMMARY, CONCLUSION AND FUTURE WORK...............................................74
5.1 Summary and Conclusions.............................................................................74
5.2 Future Work....................................................................................................75

REFERENCES........................................................................................................76

APPENDIX.............................................................................................................78
LIST OF FIGURES

Figure 1.1 Block diagram of a basic communication system.................................10
Figure 1.2 Block diagram illustrating successive interference cancellation..............13
Figure 2.1 Signal space diagram for QPSK and OQPSK........................................16
Figure 2.2 QPSK modulator...................................................................................18
Figure 2.3 Example sequences of I and Q bits of OQPSK, illustrating time offset...20
Figure 2.4 Block diagram of one type of MSK transmitter....................................22
Figure 2.5 MSK waveform.....................................................................................24
Figure 2.6 Normalized power spectral density $G(f)$ vs. normalized frequency offset from carrier, $(f-f_c)/R$ for QPSK, OQPSK and MSK..................................................26
Figure 2.7 Illustration of two-user wireless communication system that can result in unequal received signal strengths at a base station.................................28
Figure 2.8 Block diagram of the system model.....................................................29
Figure 2.9 Block diagram of the interference cancellation set-up........................32
Figure 3.1 Block diagram of amplitude estimation procedure.............................36
Figure 3.2 Model of the receiver............................................................................37
Figure 3.3 Correlation of the baseband signal $v_i(t)$ with training bits in the I and Q channel.............................................................................................................39
Figure 3.4 Block of 2N bits with the training bits distributed at the two ends..........................................................42
Figure 3.5 Sinusoidal pulse with sampling instant $\tau$ shown...............................56
Figure 4.1 Plot of amplitude vs. block size $N$ for different values of $E_b/N_0$ (3, 6 and 10 dB). True amplitude, absolute value, and rms estimate shown..........................65
Figure 4.2 NMSE vs. block size $N$, single user, rectangular pulses for $N_s=1$ and 2 samples / symbol, and different values of $E_b/N_0$: 3, 6 and 10 dB.........................66
Figure 4.3 NMSE vs. $E_b/N_0$, single user, rectangular pulses for $N=50$. Full MLSE, absolute value, and rms estimates.................................................................67
Figure 4.4 NMSE vs. $N$, single user, sinusoidal pulses, several values of $E_b/N_0$: 3, 6 and 10dB. ‘rms’ estimate also shown.................................................................68
**Figure 4.5** NMSE vs. N, rectangular pulses, two users and two conditions: bit and phase synchronous, and bit and phase asynchronous, for different values of $E_b/N_0$ and NFR....69

**Figure 4.6** NMSE vs. N, sinusoidal pulses, two users and two conditions--bit and phase synchronous, and bit synchronous and phase asynchronous, for different values of $E_b/N_0$ and NFR………………………………………………………………………………….70

**Figure 4.7** NMSE vs. N, sinusoidal pulses, two users and two conditions--bit asynchronous and phase synchronous, and bit and phase asynchronous, for different values of $E_b/N_0$ and NFR……………………………………………………………………………...……….71

**Figure 4.8** Plot of NMSE vs. NFR for users (phase and bit asynchronous) for different values of $E_b/N_0$: 2, 4, and 6 dB. Plot using exact values of amplitude of strong signal also shown…………………………………………………………………………...………..73
Chapter 1
Introduction

In this chapter, we give a brief description of a basic communication system. We also discuss the features of minimum shift keying (MSK) and its applications such as in GSM [1]. We also discuss the utility of interference cancellation and the different techniques employed for it. In addition, we introduce the maximum likelihood estimator and a statistical measure of estimator quality: the normalized mean square error. We end the chapter with a description of the scope of the thesis.

1.1 Background-Basic Communication System

The advent of digital communication systems was expedited due to many factors, including the growing need for data communication and the widespread production of digital circuitry. Digital circuits are less subject to distortion and some interference than are analog circuits. These digital circuits are hence more reliable and can also be produced at a lower cost. A typical block diagram of a communication system is shown in Figure 1.1 [2]
The source encoding is performed primarily to remove redundant information while the channel coding is done in order to decrease the probability of bit error in the presence of random noise. Modulation is necessary to convert the symbols to waveforms that are compatible with the transmission channel. The frequency spreading produces a signal which is less vulnerable to interference, thereby enhancing the privacy of the signal, and in many systems, is optional, or not used. The lower blocks describe the signal transformations from the receiver to the sink and essentially reverse the steps performed by the upper blocks.

1.2 Applications of MSK-GSM & IC

MSK is a popular signaling technique due to its compact spectrum, constant envelope and good performance. It is used in numerous applications,
including the well known Global System for Mobile Communications (GSM) [1] (which uses a filtered version, Gaussian MSK), and in frequency hopped spread spectrum systems, and other frequency division multiplexing (FDM) arrangements. GSM uses a filtered version of MSK (GMSK) wherein a Gaussian pre modulation filter smoothes the phase trajectory of the MSK signal to limit the instantaneous frequency variations. In applications such as GSM or multibeam satellite FDM, where substantial co-channel interference (CCI) is present, the CCI can be simply tolerated as a limitation to performance and capacity, or actively mitigated. Techniques for mitigating CCI include antenna diversity and interference cancellation (IC) [3].

For effective IC, an accurate estimate of the amplitudes of the interfering signal(s) is required. Amplitude estimation in the presence of CCI is the subject of this work. We consider sampled received signals, and unknown interfering data sequences. Our focus is on maximum likelihood estimators for the good performance they can provide [4], [5],[6]. The maximum likelihood estimate is an unbiased minimum variance estimator for large data set size, and is often preferred over similar estimates such as least-squares estimates, because of its comparatively smaller asymptotic variance [7]. Where analysis becomes intractable, we resort to approximations. These approximations are based upon observations and performance of progressively more complex cases, leading up to our case of interest, which is amplitude estimation of MSK in the presence of a single co-channel interferer.
In minimum shift keying, bandlimiting the signal does not cause the envelope to become zero as there are no abrupt changes in phase at bit transition instants. As the amplitude is kept constant, the MSK signals can be amplified using efficient non-linear amplifiers. The continuous phase property of the MSK enables it to be used for highly reactive loads.

1.3 Interference Cancellation

Co-channel interference refers to the presence of an interfering signal appearing within the desired signal’s bandwidth. The interfering signal may originate due to other authorized users in the spectrum, accidental transmissions, radiation splitting from antenna side-lobes, etc. Interference cancellation (IC) is instrumental in improving system performance in CCI. One popular IC technique—successive interference cancellation, or SIC—works by canceling interfering signals by successively detecting and canceling each interferer, beginning with the strongest received signal, or the signal with the largest received power.
Figure 1.2: Block diagram illustrating successive interference cancellation

The basic interference cancellation diagram is shown in Figure 1.2. The strongest signal is first demodulated to help isolate the strong signal from the rest of the signals in the channel. A second version of the original signal, delayed in order to make up for the strong signal demodulator delay, is used for IC. The remodulated strong signal is subtracted from the delayed received signal, thus leaving just the signals of lesser strength. This process is repeated and the interfering signals are removed in decreasing order of their strength. As noted, this general technique is known as successive interference canceling (SIC) [3].
1.4 Thesis Scope

In this thesis, we examine the problem of amplitude estimation in the presence of co-channel interference. In multi-user systems, co-channel interference (CCI) can present the major impediment to performance in terms of error probability and capacity. We consider the maximum likelihood estimation technique for amplitude estimation, for the practical case of sampled waveforms. We assume a single interfering user and derive the full ML estimator as a function of the number of samples per symbol, and the sampling phase.

To begin, we review analysis for the corresponding rectangular pulse case for insight and also consider the effects of both bit and carrier phase synchronism on performance. In order to quantify the performance of our estimators, we make use of a statistical quantity—the normalized mean square error (NMSE)—to compare the performance under different assumptions regarding bit and carrier phase synchronism. Using high signal to noise approximations, we show that the ML estimator of MSK becomes a scaled version of that used for rectangular pulses, with the scaling dependent on the sampling rate and sampling phase. We also show that our approximate estimators perform well, and are attractively simple in form.
Chapter 2
Signal and System Models

In this chapter, we discuss the characteristics of various digital signaling techniques, including *quadrature phase shift keying* (QPSK), *offset quadrature phase shift keying* (OQPSK) and *minimum shift keying* (MSK). We review the advantages of each technique and compare their probability of bit error ($P_b$) performance and their power spectra. We then describe the system model employed and the steps involved in estimating the amplitude of the strong signal in the channel. We also discuss an application of the likelihood estimate--interference cancellation--where we estimate the probability of bit error for the weak signal for different values of SNR of the weak signal.

2.1 Phase Modulations

The increased demand for digital transmission channels has required the maximizing of bandwidth efficiency. This is the foremost reason for employing spectrally efficient modulation techniques. Use of spectrally efficient modulations helps lessen the spectral congestion problem in various communication channels. In many cases, such as the case of satellite systems, we have additional modulation requirements. One such requirement is a constant envelope modulation technique, which is necessary for the good performance in the presence of nonlinear transponders. The absence of the constant envelope modulation would give rise to extraneous sidebands when transmitting a signal with amplitude fluctuations. This phenomenon is sometimes known as “spectral re-growth.” These sidebands, if present, would deprive the information signals of some
of their portion of transponder power and cause adjacent channel interference or co-channel interference (interference with other communication channels). Offset quadrature phase shift keying (OQPSK) and minimum shift keying (MSK) are two popular modulation schemes that are often employed for systems using nonlinear transponders. We focus upon these (4-ary) phase modulations in this work.

2.1.1 QPSK Signaling

In quadrature phase shift keying (QPSK) modulation, a sinusoidal carrier is varied in phase while maintaining a constant amplitude and frequency. The term "quadrature" implies that there are four possible phases (4-PSK) during each symbol time which the carrier can assume, as shown in the signal space diagram below in Figure 2.1.1 [2]

![Figure 2.1: Signal Space diagram for QPSK and OQPSK](image_url)
Let the input bipolar pulse sequence be represented as the waveform $d(t)$:

$$d(t) = \sum_k d_k p(t - kT)$$

where $d_k \in \{\pm 1\}$ represents the binary one and zero, and $p(t)$ is a basic (often rectangular) pulse shape. The time $T$ is the input bit duration. The bit duration is defined as the time taken by a bit to pass a point in the transmission link. For QPSK, the pulse stream is divided into an in-phase stream $d_I(t)$ and quadrature stream $d_Q(t)$, which are represented using the following subsequences:

$$d_I = d_{0}, d_{2}, d_{4}, d_{6} \ldots \ldots$$
$$d_Q = d_{1}, d_{3}, d_{5}, d_{7} \ldots \ldots$$

(2.1)

where each subsequence has half the rate of $d_k(t)$. The bit rate is the rate at which the bits are transmitted across the communication channel and is expressed in bits/second. Similarly, the symbol rate is defined as the rate at which the pairs of bits (symbol) is transmitted across the channel and is expressed in symbols/second. The QPSK modulator shown in Figure 2.1.2 [2] uses the sum of cosine and sine terms. The pulse stream $d_I(t)$ amplitude modulates the cosine function with amplitude +1 or -1, thereby shifting the phase of the cosine function by 0 or $\pi$. This results in a BPSK waveform on the “in-phase,” or “I” channel. The pulse stream $d_Q(t)$ similarly modulates the sine waveform, and likewise yields a BPSK waveform, orthogonal to the cosine function.
Amplitude modulating the in-phase and quadrature data streams onto the cosine and sine functions of the carrier wave we get the transmitted waveform equation as follows:

\[ s(t) = \frac{1}{\sqrt{2}} \cos(\omega_0 t + \pi/4) \]

\[ s(t) = \cos(\omega_0 t + \theta(t)) \] (2.2)

This can also be represented as

\[ s(t) = \sqrt{\frac{2E_b}{T_s}} \left[ \frac{1}{\sqrt{2}} d_I(t) \cos \left( 2\pi f_0 t + \frac{\pi}{4} \right) + \frac{1}{\sqrt{2}} d_Q(t) \sin \left( 2\pi f_0 t + \frac{\pi}{4} \right) \right] \] (2.3)
where $\theta(t) = 0^\circ, \pm 90^\circ, 180^\circ$ during any symbol time $T_s$, depending on the combination of $d_i(t)$ and $d_o(t)$.

We know that the union bound provides an estimate of the average probability of bit error for a particular modulation signal, $P_s(e \mid s_i)$ given as [8]

$$P_s(e \mid s_i) \leq \sum_{j=1}^{\infty} Q\left(\frac{d_{ij}}{\sqrt{2N_o}}\right)$$

(2.4)

where the complementary error function $Q(x)$ is defined as

$$Q(x) = \int_x^\infty \frac{1}{\sqrt{2\pi}} \exp(-x^2/2)dx$$

and $d_{ij}$ is the Euclidean distance between the $i^{th}$ and $j^{th}$ signal points in the constellation. From Figure 2.1.1, it can be observed that the distance between the adjacent points is $\sqrt{2E_s}$ and $E_s = 2E_b$ since each symbol consists of 2 bits. Here $E_s$ and $E_b$ represent the symbol and bit energies, respectively. Thus the distance between the adjacent points in the constellation is $2\sqrt{E_b}$. Thus using (2.4), we get

$$P_{e,QPSK} = Q\left(\frac{2E_b}{\sqrt{N_0}}\right)$$

(2.5)

It turns out that (2.5) represents the probability of bit error for QPSK and MSK also, as discussed below.
2.1.2. OQPSK Signaling

The staggered QPSK or OQPSK signaling scheme differs from QPSK signaling in the way in which the in-phase and quadrature baseband waveforms are time aligned. In QPSK, the synchronously aligned even and odd pulse streams are both transmitted at the same rate, expressed as \( R_s = (1/2T) \) symbols/s so that their transitions coincide. However, in the case of OQPSK, the alignment of the pulse streams \( d_i(t) \) and \( d_q(t) \) is offset by \( T \) as shown in Figure 2.1.3.

![Figure 2.3. Example sequences of I and Q bits of OQPSK, illustrating time offset](image)

Hence, in OQPSK, the two pulse streams can not change states simultaneously. The primary advantage of OQPSK over QPSK is that the envelope will change but never
become zero during a transition. This is because of the absence of phase transition of $180^0$ in OQPSK signaling. The phase transitions are either $0^0$ or $\pm 90^0$ every $T$ seconds. The envelope will “droop” at the region of a $\pm 90^0$ phase transition if intersymbol interference is present, e.g., when the signal undergoes bandlimiting. However the envelope is never zero due to the absence of the phase transitions of $\pm 180^0$.

### 2.1.3. MSK Signaling

Minimum shift keying is an alternative scheme where we have a continuous phase modulation. It can be viewed either as a special case of OQPSK with a sinusoidal weighting of $\sin(\pi T/2)$ (as shown in (2.7) below), or as a special case of continuous-phase frequency shift keying with a frequency separation of half the bit rate [9]. The peak frequency deviation is equal to one-fourth the bit rate $R_b$. The tone spacing in MSK is half of that used in noncoherently demodulated orthogonal FSK. A modulation index of 0.5 corresponds to the minimum frequency spacing that allows two FSK signals to be coherently orthogonal and hence the name minimum shift keying. The modulation index is defined as $k = 2\Delta F / R_b$ where $\Delta F$ is the peak RF frequency deviation and $R_b$ is the bit rate.

The MSK transmitter is illustrated in Figure 2.1.4 [10]. Here, $d_t(t)$ and $d_Q(t)$ assume the values of $\pm 1$ in symbol intervals of length $T_s = 2T_b$ seconds. The binary data streams are weighted by a cosine or a sine waveform as shown. The modulated signal can be written as
\[ X_e(t) = A \cos[w_c t - \theta_i(t)] \]

where

\[ \theta_i(t) = \tan^{-1}\left[ \left( \frac{d_Q(t)}{d_I(t)} \right) \tan(\omega_l t) \right] \]

and \( \omega_l \) is the radian frequency of the weighting functions, i.e., \( \pi/T \). If \( d_I(t) = d_Q(t) \), and the successive bits of the data stream are the same, then \( \theta_i(t) = \omega_l(t) \), whereas if \( d_I(t) = -d_Q(t) \), i.e., if the successive bits in the data stream are different, then \( \theta_i(t) = -\omega_l(t) \).

Figure 2.4: Block diagram of one type of MSK transmitter.
MSK is spectrally efficient and is employed frequently in mobile communication systems. It is preferred over other modulation techniques due to its constant envelope, spectral efficiency, self-synchronizing capability, and good BER performance.

MSK can be thought of as a special form of OQPSK where the baseband rectangular pulses are replaced with sinusoidal pulses, i.e., the transmitted waveform is

\[
s(t) = A \left[ d_I(t) \cos\left(\frac{\pi}{2T_b}t\right) \cos(2\pi f_0 t) + d_Q(t) \sin\left(\frac{\pi}{2T_b}t\right) \sin(2\pi f_0 t) \right]
\]  \hspace{1cm} (2.6)

where \(d_I(t)\) and \(d_Q(t)\) are similar to the waveforms represented in (2.1), and \(f_0\) is the carrier frequency. This kind of representation is referred to as precoded MSK, and this will be explained subsequently.

MSK can also be represented as a special case of continuous phase frequency shift keying (CPFSK) with transmitted signal

\[
s(t) = A \cos\left[ 2\pi \left( f_0 + \frac{d_k}{4T} \right) t + x_k \right] \hspace{1cm} kT < t < (k+1)T
\]  \hspace{1cm} (2.7)

where \(d_k \in \{\pm 1\}\) represents the \(k^{th}\) bipolar data bit and \(x_k\) is the phase constant during the \(k^{th}\) bit interval. In quadrature form this becomes

\[
s(t) = a_k \cos\left(\frac{\pi}{2T_b}t\right) \cos(2\pi f_0 t) - b_k \sin\left(\frac{\pi}{2T_b}t\right) \sin(2\pi f_0 t)
\]  \hspace{1cm} (2.8)
where \(a_k = \cos x_k\), and \(b_k = d_k \cos x_k\). The symbol weighting functions \(\cos(\pi/(2T))\) and \(\sin(\pi/(2T))\), for the I and Q components, respectively, are half-cycle sinusoidal pulses of duration \(2T\) seconds.

An example MSK waveform is shown in Figure 2.1.5. This waveform is based on the representation of (2.6) and is plotted for the data sequence \(d_i = 1, -1, -1, 1, \ldots\) and \(d_k = -1, 1, -1, -1, \ldots\). The in-phase and quadrature phase components are weighted by their
respective sinusoidal weighting functions and their sum yields the MSK waveform shown. It can be seen that the transition occurring at $t=0.25$, where there is a bit transition, is smooth. It is also observed that the waveform maintains constant phase and continuous envelope.

### 2.1.4. Power Spectra

When the input data is an uncorrelated sequence, the RF power spectrum is obtained by frequency shifting the magnitude squared of the Fourier transform of the baseband pulse-shaping function [8]. In the case of MSK, the baseband pulse shaping function is given as

$$p(t) = \begin{cases} \cos \left( \frac{\pi}{2T} t \right) & |t| < T, \\ 0 & \text{otherwise}. \end{cases}$$

The power spectral density of MSK is given by

$$G(f) = \frac{16PT}{\pi^2} \left( \frac{\cos(2\pi f t)}{1 - 16f^2T^2} \right)^2 [9],$$

whereas the power spectral density of QPSK and OQPSK is given by

$$G(f) = 2PT \left( \frac{\sin \frac{2\pi f T}{\pi T}}{2\pi f T} \right)^2.$$  

The power spectral densities of MSK, QPSK and OQPSK are shown in Figure 2.6.
It is observed that the MSK spectrum has lower side lobes than QPSK and OQPSK. Approximately ninety nine percent of the MSK power is contained within a bandwidth $B=1.2/T$, whereas the 90% power bandwidth of QPSK and OQPSK is $B=8/T$. The smoother pulse functions used in MSK are responsible for the faster spectral “roll-off” with frequency. It is also observed that the main lobe of MSK is wider than that of QPSK and OQPSK.
QPSK and OQPSK, implying that by the main lobe width measure, it is less spectrally efficient than the QPSK techniques. However, since there is no abrupt phase change at the bit transition periods, bandlimiting the MSK signal with additional filtering does not result in the envelope becoming zero. This bandlimiting is required in essentially all practical systems with multiple frequency division multiplexed (FDM) channels. Hence when filtered, the MSK spectral efficiency improves without significantly degrading performance. Due to the constant amplitude, the MSK signals can be amplified using efficient nonlinear amplifiers. The MSK scheme also has simple demodulation and synchronization circuits making it a preferred modulation scheme for mobile radio communications.

2.2. Problem Setting

Figure 2.2.1 illustrates a situation in which two transmitters, $T_{x1}$ and $T_{x2}$, are simultaneously transmitting signals to a central base station. This can represent for example a terrestrial cellular radio setting. Often the base station has an omni-directional antenna pattern. When one transmitter is located closer to the base station than others, and the transmit powers are nearly equal, the signal received from the closer transmitter ($T_{x1}$) is stronger than the signal received from farther transmitter ($T_{x2}$).
Figure 2.7 Illustration of two-user wireless communication system that can result in unequal received signal strengths at a base station.

Generally, in mobile systems, transceiver locations can vary over a nearly unlimited number of spatial positions. For illustration of our application, Figure 2.2.1 shows the first transmitter located closer to the receiver. For a given constant value of mobile transmitter power, the received signal from transmission by the closer mobile is stronger. (We ignore the use of transmitter power control here, for simplicity—when the transmissions emanate from non-cooperative users, or from users in different cells, power
control will not apply.) The second transmitter is located farther away from the base station and we hence model the strength of its signal at the receiving base station as less than that of the first signal.

### 2.3 System Model

The block diagram representing the system model which we use throughout our project is illustrated in Figure 2.3.1

![Block diagram of the system model.](image)

We consider the case where the signal of interest is coherently down-converted, for simplicity. In this case, amplitude estimation is done on either the in-phase \((I)\) or quadrature \((Q)\) channel, or both. We use identical processing on both channels. For
packet-based systems that use some known training bits, phase estimation can be achieved “offline” via simple correlation with the training sequence [1] and subsequent phase rotation, after which amplitude estimation proceeds. We assume perfect frequency coherence between the received signal and the receiver local oscillators, as estimation and correction of small Doppler shifts can be achieved with the same training sequence correlation as used for phase estimation. For the cases where co-channel signals are present, we assume the receiver has determined both the frequency and phase of the stronger signal. We also allow the signals to be timing asynchronous to each other. A detailed discussion of these issues is provided in chapter 3.

Amplitude estimation on the stronger signal without any known data can represent the case where the co-channel interferer is stronger than the desired signal, in which case frequency/phase corrections are made using decision-direction. We begin with the simplest case possible, for which the closed-form ML amplitude estimator solutions can be obtained. For more complex conditions, we employ approximations and computer simulations to study the effects of phase non-coherence and timing asynchronism between the two co-channel signals.

As illustrated by the block diagram in Figure 2.3.1, the system model consists of two independent transmitters. The signal transmitted by transmitter 2 is assumed to be of lesser strength than that of the first. In a practical case, we also have an associated delay and phase change of each signal.

As an application of amplitude estimation, we consider interference cancellation. In this application, the aim is to demodulate the stronger received signal, reconstruct it, then subtract it from the total received signal, then proceed with demodulation of the
weaker signal [3]. The estimate of the stronger signal is first obtained, then followed by collection of the $I$ and $Q$ parts of the demodulated version of the signal. The signal is then re-modulated. The re-modulated signal is then subtracted from a delayed version of the original received signal in order to allow detection of the weak signal. For the remodulated signal, we also require phase alignment. The bit error probability of the weaker signal can hence be obtained by demodulating the resulting signal corrupted with AWGN.

We use an additive white Gaussian noise (AWGN) channel. By this we mean that the only impairment in the signal transmission is the linear addition of additive white Gaussian noise with constant spectral density. Other realistic channel effects like fading, frequency selectivity, and dispersion are absent. However, thermal noise caused by the thermal motion of electrons in the dissipative components, cannot be eliminated. The thermal noise is modeled as a zero mean Gaussian random process. The Gaussian probability density function is given by

$$p(n) = \frac{1}{\sqrt{2\pi}\sigma^2} \exp\left[-\frac{n^2}{2\sigma^2}\right]$$

(2.9)

where the argument $n$ denotes the random Gaussian variate, a sample of the continuous time Gaussian random process. The noise is zero-mean Gaussian with variance $\sigma^2$, represented as $n \sim \eta(0, \sigma^2)$ and the $k^{th}$ channel output $r_k$, obtained by sampling the received downconverted signal at time $kT_s$, is a real number. The probability density of the output conditioned upon symbol $x_k$ at the input is Gaussian in form, specifically given by

$$p_{y|x}(y|X = x_k) = \frac{1}{\sqrt{2\pi}\sigma^2} \exp\left[-\frac{(y-x_k)^2}{2\sigma^2}\right].$$

It is clear from the expression that
the channel output at any instant \( k \) depends only on the input at the instant \( k \) and not on the earlier (or later) inputs, i.e., the effect on the detection process of an AWGN channel is that the noise affects each transmitted symbol independently. We hence refer to this channel as a memoryless channel.

### 2.3.1 Interference Cancellation

The concept of interference cancellation and its usefulness in a communication system were described in the previous section. The system shown below in Figure 2.3.1 is used to estimate the probability of bit error in the weak signal in the system. In our system, we consider two users with signals \( s_1 \) and \( s_2 \) using the same communication channel. The received signal is represented as \( r(t) = s_1(t) + s_2(t) + n(t) \), where \( s_1 \) represents the stronger of the 2 signals and \( n \) is AWGN.

![Block diagram of the interference cancellation set-up.](image)

Figure 2.9: Block diagram of the interference cancellation set-up.
The $I$ and $Q$ channel signal estimates are used to find the amplitude estimate of the strong user using the 1st set of detectors. The bit estimates—assumed to have reasonably low bit error rate—are demodulated and recovered. The reconstructed strong signal is then subtracted from the delayed version of the original signal in order to enable us to obtain only the weak signal in the presence of noise, to which we may then apply the same detection technique. The interference cancellation technique can be a reliable method to detect the weak signal. Thus, we apply advanced receiver processing to enable higher aggregate throughputs but costing us complexities in hardware and software.

The amplitudes of the strong and weak signal are related using the term near-far ratio ($NFR$). The $NFR$ in dB is represented as

$$NFR(dB) = 10 \log \left( \frac{P_1}{P_2} \right)$$  \hspace{1cm} \text{(2.10)}$$

where $P_1$ and $P_2$ are the received powers of the signals 1 and 2. However, since

$$\left( \frac{P_1}{P_2} \right) \approx \left( \frac{A_1}{A_2} \right)^2,$$

we can express the $NFR$ in terms of amplitudes as

$$NFR(dB) = 20 \log \left( \frac{A_1}{A_2} \right).$$
Chapter 3
Amplitude Estimation

In this chapter, we discuss the assumptions made in the two-user case for our development of the estimator of the amplitude of the strong signal. We employ a maximum likelihood (ML) estimator and other types of estimators. The ML estimator is known to be the minimum variance unbiased estimator when the pdf is known. Due to approximations and for practical application, the estimators we derive will not be exactly ML, but will be shown to perform well. We obtain the (ML, or near-ML) amplitude estimator for four different cases:

1. rectangular pulses in additive white Gaussian noise (AWGN)
2. sinusoidal pulses in AWGN
3. rectangular pulses in AWGN and a single co-channel interferer.
4. sinusoidal pulses in AWGN and a single co-channel interferer.

The estimators’ statistical quality is expressed in terms of normalized mean-square error, commonly used in SNR estimators, e.g., [6]. We also assess the resulting estimate quality in terms of bias and consistency.

3.1. Assumptions-Frequency Coherence

A bandpass signal transmitted over a communication link can be expressed as...
\[ s(t) = [I(t) \cos(\omega_c t) + Q(t) \sin(\omega_c t)]A = \text{Re}\{\tilde{s}(t)e^{j\omega_c t}\} \quad (3.1) \]

where

\[ \tilde{s}(t) = [I(t) - jQ(t)]A; \quad (3.2) \]

and \( A = \sqrt{2E_b/T_b} \). For MSK, the in-phase and quadrature terms are expressed as

\[ I(t) = \sum_{k=1}^{N} I_k p(t - kT_s) \]
\[ Q(t) = \sum_{k=1}^{N} Q_k p(t - kT_s - T_s/2) \]

where \( p(t) \) is a pulse shape (defined subsequently) and the symbols \( I_k, Q_k \in \{\pm 1\} \).

A flow diagram of the amplitude estimation procedure is illustrated in Figure 3.1.
Figure 3.1. Block diagram of amplitude estimation procedure.
Generally, \{I_k\} and \{Q_k\} result from the serial to parallel (S:P) conversion of an input data sequence. Hence the I and Q bits are either the even-indexed or odd-indexed bits, respectively. The sinusoidal pulse shape is

\[
p(t) = \begin{cases} 
\sin(\pi / T_s) & 0 \leq t \leq T_s \\
0 & \text{else}
\end{cases}
\] (3.3)

The signals are also assumed to undergo a delay in traversing the additive white Gaussian noise (AWGN) channel. The received signal for a single user is hence represented as \( r(t) = s(t - \tau) + n(t) \). The delay \( \tau \) due to the channel propagation time in the non-dispersive channel is modeled as random, and the received signal is

\[
r(t) = I(t - \tau) \cos(\omega_c t - \theta) + Q(t - \tau) \sin(\omega_c t - \theta) + n(t) (3.4)
\]

where \( \theta = \omega_c \tau \). The diagrammatic representation of the receiver is shown in Figure 3.2.

![Figure 3.2: Model of the receiver](image)
The resulting baseband components after down conversion are denoted $v_r(t)$ and $v_i(t)$, and these can be expressed as a complex signal $v(t)=v_r(t)+jv_i(t)$, whose real and imaginary parts are

$$v_r(t) = A[I(t-\tau)\cos \theta - Q(t-\tau)\sin \theta]$$ (3.5a)

$$v_i(t) = A[I(t-\tau)\sin \theta + Q(t-\tau)\cos \theta]$$ (3.5b)

If we include in the above expression a Doppler shift of the carrier frequency originating from a constant velocity $v$, we obtain

$$v_r(t) = A[I(t-\tau)\cos(\pm 2\pi f_D t - \theta) + Q(t-\tau)\sin(\pm 2\pi f_D t - \theta)]$$ (3.6a)

$$v_i(t) = A[-I(t-\tau)\sin(\pm 2\pi f_D t - \theta) + Q(t-\tau)\cos(\pm 2\pi f_D t - \theta)]$$ (3.6b)

where the $\pm f_D$ indicates a positive or negative Doppler shift. If $f_D=0$, we obtain (3.5).

For simplicity, let us first consider the case of $f_D=0$. Then for coherent detection the receiver must estimate the channel phase $\theta$ in (3.5). This phase can be estimated by correlating $v_r$ and $v_i$ with known data. Nearly all systems send some amount of known data for such purposes, and for other system-level operations such as user identification and packet frame boundary delineation. The baseband components in (3.5) are hence correlated with the known bits in the $I$ and $Q$ channel as shown for $v_i$ in Figure 3.3.
The normalized correlation coefficients that result from these operations are defined as

\[
\rho_{II} = \frac{1}{N_t} \sum_{i=1}^{N_t} I_i I_{ni} \tag{3.7a}
\]

\[
\rho_{IQ} = \frac{1}{N_t} \sum_{i=1}^{N_t} I_i Q_{ni} \tag{3.7b}
\]

We perform similar operations on \(v_q(t)\), yielding \(\rho_{QQ}\) and \(\rho_{QI}=\rho_{IQ}\).

In ideal cases, (3.7a) equals unity and (3.7b) is a comparatively negligible quantity \(\approx 0\), i.e., the cross correlation is zero. For example, with a training sequence of length \(N_t=20\) bits, random sequences yield an average crosscorrelation of approximately
\[ \frac{1}{\sqrt{N_t}}, \text{ which for } N_t=20 \text{ is approximately } 13 \text{ dB down from the autocorrelation peak value. Hence, correlating } v_r(t) \text{ and } v_i(t) \text{ with the } I \text{ channel bit stream yields primarily the } I \text{ portion of the signal and we similarly obtain the } Q \text{ portion of the signal by correlating } v_r(t) \text{ and } v_i(t) \text{ with the } Q \text{ channel bit stream. Thus we have} \]

\[
v_n \equiv A \rho_{II} \cos(\theta), \quad v_{rQ} \equiv -A \rho_{QQ} \sin(\theta), \quad v_{iu} \equiv A \rho_{II} \sin(\theta), \quad v_{iq} \equiv A \rho_{QQ} \cos(\theta)\]

We then add these as follows:

\[
v_n + v_{iq} \equiv 2A \rho_{II} \cos(\theta) \quad \text{(3.8a)}
\]

\[
-v_{rQ} + v_{iu} \equiv 2A \rho_{II} \sin(\theta) \quad \text{(3.8b)}
\]

from which we obtain an approximation to \( \tan(\theta) \) by dividing (3.8b) by (3.8a). The estimate of the angle \( \theta \) is then obtained by performing an inverse tangent (typically via a table look-up). With this estimate of \( \theta \) we can “phase rotate” (3.5) to obtain

\[
\hat{I}(t - \tau) = v_r(t) \cos \hat{\theta} + v_i(t) \sin \hat{\theta} = AI(t - \tau) \quad \text{(3.9a)}
\]

\[
\hat{Q}(t - \tau) = -v_r(t) \sin \hat{\theta} + v_i(t) \cos \hat{\theta} = AQ(t - \tau) \quad \text{(3.9b)}
\]

which corresponds to phase rotating \( v_r - jv_i \).

To illustrate this, using (3.5a) and (3.5b) in (3.9a) and (3.9b), respectively, we get

\[
\hat{I}(t - \tau) = AI(t - \tau) \cos \theta \cos \hat{\theta} - AQ(t - \tau) \sin \theta \cos \hat{\theta} + \\
AI(t - \tau) \sin \theta \sin \hat{\theta} + AQ(t - \tau) \cos \theta \sin \hat{\theta}
\]

\[
\hat{Q}(t - \tau) = -AI(t - \tau) \cos \theta \sin \hat{\theta} + AQ(t - \tau) \sin \theta \sin \hat{\theta} + \\
+ AI(t - \tau) \sin \theta \cos \hat{\theta} + AQ(t - \tau) \cos \theta \cos \hat{\theta}
\]

or

\[
\hat{I}(t - \tau) = AI(t - \tau) \left[ \cos \theta \cos \hat{\theta} + \sin \theta \sin \hat{\theta} \right] + AQ(t - \tau) \left[ \sin \hat{\theta} \cos \theta - \sin \theta \cos \hat{\theta} \right] \quad \text{(3.10a)}
\]
\[
\hat{Q}(t - \tau) = AI(t - \tau)\left[\sin \theta \cos \hat{\theta} - \cos \theta \sin \hat{\theta}\right] + AQ(t - \tau)\left[\sin \hat{\theta} \sin \theta + \cos \theta \cos \hat{\theta}\right]
\] (3.10b)

In (3.10), if we assume the estimate to be nearly perfect \((\theta \approx \hat{\theta})\), then using basic trigonometric identities, we have

\[
\hat{I}(t - \tau) = AI(t - \tau) \tag{3.11a}
\]
\[
\hat{Q}(t - \tau) = AQ(t - \tau) \tag{3.11b}
\]

In practice, the phase estimate will not be perfect, but (3.11) will provide a good approximation. These equations (3.11) are what we use as inputs to the amplitude estimation scheme.

The above equations pertain to the case when we assume \(f_D = 0\). However, in the most practical cases, we need to account for a Doppler shift. To obtain the Doppler estimate, let us assume \(I_k(t)\) and \(Q_k(t)\) are the known data for the \(I\) and \(Q\) channels, where the index \(k\) indicates the symbols in the transmitted packet corresponding to the known (or, training) symbols. Figure 3.4 represents the block of \(I\) and \(Q\) bits each of length \(N\) containing training bits of length \(2N_f\) each.
The correlation or the dot product (analogous to (3.7)), given as $I_k Q_k^T$, is very small by design, since the training sequences are selected to have low cross-correlation (ideally zero). We assume $f_D$ to be small so that less than one cycle of $\cos(2\pi f_D t)$ appears over a single packet duration. For even relatively large velocities and moderate to long packets, this assumption is very good. With these assumptions, we can estimate the Doppler frequency via

$$f_D \approx \frac{\Delta \phi}{\Delta t}$$

where $\Delta \phi = \phi_2 - \phi_1$. Each term in $\Delta \phi$ is an estimate of the phase over one portion of the packet and corresponds to the estimate of $\theta$ discussed earlier.

If $f_D > 0$, we have
\[ v_r(t) = A[I(t - \tau)\cos(2\pi f_D t - \theta) + Q(t - \tau)\sin(2\pi f_D t - \theta)] \]
\[ v_i(t) = A[-I(t - \tau)\sin(2\pi f_D t - \theta) + Q(t - \tau)\cos(2\pi f_D t - \theta)] \]  \hspace{1cm} (3.12)

whereas if \( f_D < 0, \)
\[ v_r(t) = A[I(t - \tau)\cos(2\pi f_D t + \theta) - Q(t - \tau)\sin(2\pi f_D t + \theta)] \]
\[ v_i(t) = A[I(t - \tau)\sin(2\pi f_D t + \theta) + Q(t - \tau)\cos(2\pi f_D t + \theta)] \]  \hspace{1cm} (3.13)

We then correlate \( v_r \) with \( I_{km} \) and \( v_i \) with \( Q_{km} \)
\[ V_{\cos,m} = V_r \cdot I_{km} + V_i \cdot Q_{km} \]  \hspace{1cm} (3.14)
\[ V_{\sin,m} = -V_i \cdot I_{km} + V_r \cdot Q_{km} \]

where \( m \) denotes the position of the training bits in the block \((m=1, 2)\)

This correlation enables the estimation of the delay \( \tau \), which is required to time align \( v_r, v_i, I_{km}, Q_{km} \) to obtain \( V_{\cos,m} \) and \( V_{\sin,m} \). Let us assume that we have \( J \) samples per \( T_s \). The signals \( V_{\sin,m} \) and \( V_{\cos,m} \) are then sampled at times \( lT_s \). This is followed by “sliding” the training sequences by the received data blocks until we obtain a peak, which gives us the desired value for \( \tau \). Then we have
\[ v_{\cos,m} \approx A \sum_{l=1}^{N_f} \cos(2\pi f_D lT_s - \theta)\big[I(lT_s)I_k(lT_s) + Q(lT_s)Q_k(lT_s)\big] \]  \hspace{1cm} (3.15)
a)
\[ v_{\sin,m} \approx A \sum_{l=1}^{N_f} \sin(2\pi f_D lT_s - \theta)\big[I(lT_s)I_k(lT_s) + Q(lT_s)Q_k(lT_s)\big] \]  \hspace{1cm} (3.16)

obtained by re-aligning (3.14), where we assume the cross correlations \( \sum_l I(lT_s)Q_k(lT_s) \) and \( \sum_l Q(lT_s)I_k(lT_s) \) are much smaller than the autocorrelations \( \sum_l I(lT_s)I_k(lT_s) \) and
Thus we have $v_{c,1}$ and $v_{s,1}$ for the known sequence at the beginning of the packet given as

$$v_{c,1} \approx 2A \sum_{l=1}^{N_1} \cos(2\pi f_D l T_s - \theta)$$  \hspace{1cm} (3.17a)$$

$$v_{s,1} \approx 2A \sum_{l=1}^{N_1} \sin(2\pi f_D l T_s - \theta)$$  \hspace{1cm} (3.17b)$$

When $f_D$ is very small, so that $f_D T_s N \ll 1$, we can approximate (3.17) as

$$v_{c,1} \approx 2AN_1 J \cos(2\pi f_D N_1 JT_s - \theta)$$  \hspace{1cm} (3.18a)$$

$$v_{s,1} \approx 2AN_1 J \sin(2\pi f_D N_1 JT_s - \theta)$$  \hspace{1cm} (3.18b)$$

where the approximation of the amplitude $2AN_1 J$ is perfect in the case of a rectangular signal but is approximate for MSK pulses. This is because for MSK pulses, $2AN_1 J$ is the amplitude at the peak and would hence be larger than the actual amplitude samples of the sinusoidal pulse. Similarly we find $v_{c,2}$ and $v_{s,2}$ for the known sequences at the end of the $N$ symbol burst.

$$v_{c,2} \approx 2AN_1 J \cos(2\pi f_D N_1 JT_s - \theta)$$  \hspace{1cm} (3.19a)$$

$$v_{s,2} \approx 2AN_1 J \sin(2\pi f_D N_1 JT_s - \theta)$$  \hspace{1cm} (3.19b)$$

Then, the phase estimates are given as

$$\hat{\phi}_1 = \tan^{-1} \left( \frac{v_{s,1}}{v_{c,1}} \right) = 2\pi f_D N_1 JT_s - \theta$$  \hspace{1cm} (3.20a)$$

$$\hat{\phi}_2 = \tan^{-1} \left( \frac{v_{s,2}}{v_{c,2}} \right) = 2\pi f_D JT_s - \theta$$  \hspace{1cm} (3.20b)$$

Finally, our estimate of the Doppler frequency is
When $f_D < 0$, we obtain $\hat{f}_D = f_D < 0$. We hence consider $|\hat{f}_D|$ as $\hat{f}_D$ and further assume a $-\hat{f}_D$ instead of an $\hat{f}_D$ in the following discussion. According to the algorithm depicted in Figure 3.1, we then construct $\cos(2\pi \hat{f}_D t)$ and $\sin(2\pi \hat{f}_D t)$ and use these along with (3.9) and (3.10) to construct the following two signals

$$X_1(t) = v_1(t) \cos(-2\pi \hat{f}_D t) + v_1(t) \sin(-2\pi \hat{f}_D t)$$

$$= A[I \cos \theta - Q \sin \theta]$$

and

$$X_2(t) = v_1(t) \cos(-2\pi \hat{f}_D t) + v_1(t) \sin(-2\pi \hat{f}_D t)$$

$$= A[I \sin \theta + Q \cos \theta]$$

where (3.22b) and (3.23b) refer to the case when $\hat{f}_D = f_D$. For this case with $f_D$ not equal to zero, after construction of (3.22) and (3.23), we can estimate the phase $\theta$ by correlating $X_1$ and $X_2$ with $I_k$, to yield the following two quantities

$$s_2 = X_2 \cdot I \approx AI \cdot I \sin \theta$$

$$s_1 = X_1 \cdot I \approx AI \cdot I \cos \theta$$

This is analogous to the description provided for our earlier case with $f_D = 0$. Thus the estimate of the phase is obtained as $\hat{\theta} \approx \tan^{-1}(s_2 / s_1)$. Referring to Figure 3.1 again, the signals $y_1(t)$ and $y_2(t)$ are subsequently obtained as in (3.11)

$$y_1(t) \approx AI(t) \approx A \sum_l I_l p(t - lT_s)$$

$$y_2(t) \approx AQ(t) \approx A \sum_l Q_l p(t - lT_s)$$

(3.24)
These signals are then used in the amplitude estimator.

3.2 Maximum Likelihood Estimate (MLE)

The maximum likelihood estimate of the function depending on $k$ parameters $\theta_1, \theta_2, \ldots, \theta_k$ is obtained by choosing as estimates those values of parameters that maximize the likelihood function $L(y_1, y_2, \ldots, y_n \mid \theta_1, \theta_2, \ldots, \theta_k)$. The method of maximum likelihood estimation gives a better estimate of the parameters in the sense that the asymptotic variance of the MLE is lower than that of other comparable estimators such as the Least Squares Estimate (LSE). The LSE, generally used in curve fitting, is an optimization technique to find the best fit for a set of data to minimize the sum of the squares of the differences between the function and the data. The MLE is normally distributed and is unbiased for large sample size.

Using our sampled signal model of (2.3), the likelihood function for the $i^{th}$ received sample $r_i$ with amplitude $A$, and $N$ symbols/block is given by

$$L(A) = \prod_{i=1}^{N} p(r_i \mid A)$$

where $p(r_i \mid A)$ is the conditional pdf of the $i^{th}$ received sample. The pdf is a product form because the action of noise is independent on each symbol, hence the joint pdf of the block of samples is the product of the marginal pdfs.

The maximum likelihood estimate is obtained by maximizing the log likelihood function, which is done via conventional calculus techniques, i.e., by
setting $\frac{\partial [l(A)\]} {\partial A} = 0$, where $l(A) = \ln[L(A)]$. The natural logarithm of the likelihood function can be used because it is a monotonic function of its argument, and will not change the result found for the maximum.

### 3.2.1 Case 1: MLE for Rectangular Pulses in AWGN

In this case, the received signal $r(t) = s(t) + n(t)$, with the transmitted signal given as

$$s(t) = Ab(t) = A \sum_{i=0}^{\infty} b_i p(t - iT)$$

(3.25)

where $p(t)$ is a rectangular pulse, equal to unity from $t=0$ to $t=T$ and zero elsewhere, $T$ is the bit duration, $A$ is the signal amplitude which is to be estimated.

We first consider the single sample per bit case. In this case, the $i^{th}$ received sample is given as

$$r_i = Ab_i + n_i$$

(3.26)

where $n_i$ is the noise sample, with mean zero and variance $\sigma^2$. The expression in (3.26) holds regardless of the sampling time relative to bit transitions, due to the rectangular pulse shape. Hence the results in this case are invariant to sampling phase.

For strictly rectangular pulses, amplitude estimation can be performed without phase coherence. This is explained as follows: the transmitted signal is given as $I(t) \cos(\omega_c t) - Q(t) \sin(\omega_c t)$, and a phase shift of $\alpha$ implies that we demodulate with sinusoids having argument $\omega_c t + \alpha$. This yields an in-phase channel term...
$I(t)\cos(\alpha) - Q(t)\sin(\alpha)$, and a quadrature channel term $I(t)\cos(\alpha) - Q(t)\sin(\alpha)$. By adding the squares of these terms, we obtain $2A^2$ when the amplitude on both $I(t)$ and $Q(t)$ is $A$. In this case the estimate of $A$ is obtained simply by dividing by two, then taking a square root. We resume our ML analysis for the first case.

In order to find the likelihood function, we first express the probability density function for a binary NRZ signal as:

$$p(r_i | A, b_i) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left[ -\frac{(r_i - Ab_i)^2}{2\sigma^2} \right]$$

(3.27)

To remove the conditioning on the data bits, we note that since $b_i$ is discrete and equiprobable, its probability mass function is given as

$$p(b_i) = 0.5\delta(1-b_i) + 0.5\delta(1+b_i)$$

(3.28)

Hence we can use (3.27) and (3.28) in

$$p(r_i | A) = \int p(r_i | A, b_i) p(b_i) \, db_i$$

(3.29)

to get

$$p(r_i | A) = \frac{0.5}{\sqrt{2\pi\sigma^2}} \left[ e^{-\frac{(r_i - A)^2}{2\sigma^2}} + e^{-\frac{(r_i + A)^2}{2\sigma^2}} \right]$$

(3.30)

We will estimate using a block of $N$ symbols, for which the likelihood function $L(A)$ is given in (3.24) via independence of the noise samples, where $r = [r_1, r_2, \ldots, r_N]$. Using (3.30) in the product form of $L(A)$, we get
\[
L(A) = \left( \frac{1}{2\sigma \sqrt{2\pi}} \right)^N \prod_{i=1}^N \left\{ \exp \left[ \frac{-(r_i - A)^2}{2\sigma^2} \right] + \exp \left[ \frac{-(r_i + A)^2}{2\sigma^2} \right] \right\}
\] (3.31)

Taking a natural logarithm of (3.31), we obtain

\[
l(A) = \ln[L(A)] = \ln C + \sum_{i=1}^N \ln \left\{ \exp \left[ \frac{-(r_i - A)^2}{2\sigma^2} \right] + \exp \left[ \frac{-(r_i + A)^2}{2\sigma^2} \right] \right\}
\] (3.32)

where \( C = \left( \frac{1}{2\sigma \sqrt{2\pi}} \right)^N \). Via expansion of the exponential terms and some algebra, (3.32) becomes

\[
\ln C + \sum_{i=1}^N \ln \left\{ \exp \left[ \frac{-(r_i^2 + A^2)}{2\sigma^2} \right] \left[ \exp \left( \frac{r_i A}{\sigma^2} \right) + \exp \left( -\frac{r_i A}{\sigma^2} \right) \right] \right\}
\]

Using the definition of the hyperbolic cosine function, \( 2 \cosh x = \exp(x) + \exp(-x) \), this can be expressed in the alternative form

\[
l(A) = \ln C + \sum_{i=1}^N \left\{ \frac{-(r_i^2 + A^2)}{2\sigma^2} \right\} + \ln \left[ 2 \cosh \left( \frac{r_i A}{\sigma^2} \right) \right]
\] (3.33)

To find the maximum likelihood estimate, the partial derivative of \( l(A) \) with respect to \( A \) is taken:

\[
\frac{\partial}{\partial A} l(A) = \sum_{i=1}^N \frac{-A}{\sigma^2} + \frac{\partial}{\partial A} \left[ 2 \cosh \left( \frac{r_i A}{\sigma^2} \right) \right]
\] (3.34)
Using standard calculus identities, 
\[ \frac{\partial \ln(f(A))}{\partial A} = \frac{1}{f(A)} \frac{\partial}{\partial A} [f(A)] \]

Thus, the second term in (3.34) becomes

\[ \frac{\partial}{\partial A} \ln \left[ 2 \cosh \left( \frac{r_i A}{\sigma^2} \right) \right] = \frac{\partial}{\partial A} \left[ \ln \left( \exp \left( \frac{r_i A}{\sigma^2} \right) + \exp \left( -\frac{r_i A}{\sigma^2} \right) \right) \right] = \frac{r_i}{\sigma^2} \left[ \exp \left( \frac{r_i A}{\sigma^2} \right) - \exp \left( -\frac{r_i A}{\sigma^2} \right) \right] \]
\[ 2 \cosh \left( \frac{r_i A}{\sigma^2} \right) \]

Then using the definition of the hyperbolic sine function, 
\[ 2 \sinh x = \exp(x) - \exp(-x), \]
we have

\[ \frac{\partial}{\partial A} \ln \left[ 2 \cosh \left( \frac{r_i A}{\sigma^2} \right) \right] = \frac{r_i \sinh \left( \frac{r_i A}{\sigma^2} \right)}{\sigma^2 \cosh \left( \frac{r_i A}{\sigma^2} \right)} = \frac{r_i}{\sigma^2} \tanh \left( \frac{r_i A}{\sigma^2} \right) \]  \hspace{1cm} (3.35)

From (3.33) and (3.34), we have

\[ \frac{\partial}{\partial A} l(A) = -\frac{NA}{\sigma^2} + \frac{1}{\sigma^2} \sum_{i=1}^{N} r_i \tanh \left( \frac{r_i A}{\sigma^2} \right) \]

The maximum likelihood function, as discussed before, is obtained by equating

\[ \frac{\partial}{\partial A} l(A) \] to 0. Thus

\[ -\frac{NA}{\sigma^2} + \frac{1}{\sigma^2} \sum_{i=1}^{N} r_i \tanh \left( \frac{r_i A}{\sigma^2} \right) = 0 \]
\[ \Rightarrow N \hat{A} = \sum_{i=1}^{N} r_i \tanh \left( \frac{r_i A}{\sigma^2} \right) \]
\[ \hat{A} = \frac{1}{N} \sum_{i=1}^{N} r_i \tanh \left( \frac{r_i A}{\sigma^2} \right) \]  

(3.36)

The expression in (3.36) is simple enough, yet it contains the value \( A \) (which we are estimating) in the argument of the hyperbolic tangent, so in this form, the estimate is not exactly usable. We could substitute for \( A \) from (3.4), but then we would need both the noise sample and the data bits for processing according to (3.36). Because of this, we resort to approximations. For large values of argument, the \( \tanh \) function is approximately equal to the \( \text{sgn} \) function:

\[
\text{sgn}(x) = \begin{cases} 
1, & x > 0 \\
-1, & x < 0
\end{cases}
\]

The argument of the \( \tanh \) function is large when the signal-to-noise ratio (SNR) \( \frac{A^2}{\sigma^2} \) is large. In this case, since both \( A \) and \( \sigma^2 \) are positive, we have \( \text{sgn}[r_i A / \sigma^2] = \text{sgn}(r_i) \) and since \( r_i \text{sgn}(r_i) = |r_i| \), our approximate ML estimate of the amplitude is

\[ \hat{A} = \frac{1}{N} \sum_{i=1}^{N} |r_i| \]  

(3.37)

which is very attractive for its simplicity. For the case of \( N_S \) samples per bit, the development is completely analogous. The pdf of (3.30) becomes
where \( r_{Na} \) is a vector of \( N_s \) samples for the \( i^{th} \) bit and \( C = 0.5(2\pi\sigma^2)^{-N_s/2} \). Then the likelihood function \( L(A) \) of (3.23) is the product of \( N \) terms of the form (3.38). By the same procedure as in the single sample per symbol case, we take \( \ln[L(A)] \), then take its derivative with respect to \( A \), factor common terms in the exponentials and analogous to (3.32), obtain

\[
\hat{A} = \frac{1}{NN_s} \sum_{m=1}^{N} \sum_{m=1}^{N_s} r_m \text{tanh} \left( \frac{A}{\sigma^2} \sum_{m=1}^{N_s} r_m \right) \quad (3.39)
\]

Then by the same arguments as used to obtain (3.37), with the additional approximation that \( \text{sgn}(\sum r_m) \cong \text{sgn}(r_m) \), our high SNR approximation in the \( N_s \) samples per bit case is

\[
\hat{A} = \frac{1}{NN_s} \sum_{m=1}^{N} \sum_{m=1}^{N_s} |r_m| \quad (3.40)
\]

Using more than one sample/symbol requires faster processing, but does provide a faster convergence in terms of the number of bits (\( N \)) required achieving a given level of performance (NMSE).
3.2.2 Normalized Bias of the Amplitude Estimate

The amplitude estimate of the single user with rectangular pulses in AWGN as given in (3.37) is \( \hat{A} = \frac{1}{N} \sum_{i=1}^{N} |r_i| \). The expected value of the estimate can be bounded as follows

\[
E[\hat{A}] = \frac{1}{N} \sum_{i=1}^{N} E[|A_b + n_i|] \leq \frac{1}{N} \sum_{i=1}^{N} \{E[|A_b|] + E[|n_i|]\} = A + \frac{1}{N} \sum_{i=1}^{N} E[|n_i|] \tag{3.41}
\]

The probability density function of the AWGN sample is given in (2.9). The expected value of the absolute value of the AWGN is found by a transformation of random variables. Let \( z_i = |n_i| \) and we can derive the pdf of \( z_i \) using standard techniques. This pdf is \( f(x) = \frac{2}{\sqrt{2\pi} \sigma^2} e^{-x^2/2\sigma^2} \) where \( x \geq 0 \).

Then the expected value of \( z \) is found by

\[
E[|n|] \leq \frac{2}{\sqrt{2\pi} \sigma^2} \int_{0}^{\infty} xe^{-x^2/2\sigma^2} dx \tag{3.42}
\]

which on simplification yields

\[
E[|n|] \leq \sqrt{\frac{2}{\pi} \sigma} \tag{3.43}
\]

Hence, (3.41) can be represented as

\[
E[\hat{A}] \leq A + \frac{1}{N} \sum_{i=1}^{N} \sqrt{\frac{2}{\pi} \sigma} \leq A + \sqrt{\frac{2}{\pi} \sigma} \tag{3.44}
\]

The normalized bias of the estimate of the amplitude is defined as
\[ B_n = \frac{E(A) - A}{A} \leq \frac{\sigma}{A} \sqrt{\frac{2}{\pi}} \]  
(3.45)

or

\[ B_n \leq \sqrt{\frac{2}{\pi \text{SNR}}} , \]  
(3.46)

where SNR is the signal to noise ratio of the signal. The bound on the bias, as expected, is inversely proportional to the signal to noise ratio.

However, the exact bias can be found using the pdf of \( y = |r_i| = |Ab_i + n_i| \). The pdf in this case is given as

\[ f(x) = \frac{2}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-A)^2}{2\sigma^2}} \quad x \geq 0 \]

and the equivalent of (3.42) is given as

\[ E(y) = \int_{0}^{\infty} \frac{2y}{\sqrt{2\pi\sigma^2}} e^{-\frac{(y-A)^2}{2\sigma^2}} dy \]  
(3.47)

which on simplification yields

\[ E(y) = \frac{\sqrt{2\sigma + A}}{\sqrt{\pi}} \]  
(3.48)

which is identical to (3.44), that is, the upper bound on the bias is exact, and the normalized bias of the estimate of the amplitude is exactly \( B_n = \sqrt{\frac{2}{\pi \text{SNR}}} \). An interesting observation in this case is the lack of dependence on block size \( N \) in the expression for the bias.
3.2.3 Case 2: MLE for Sinusoidal Pulses, Bit Synchronous Case

This case represents a single MSK user, coherently downconverted. The baseband signal on either the $I$ or $Q$ channel is analogous to (3.25)

$$s(t) = Ab(t) = A \sum_{i=0}^{\infty} b_i p_s(t - iT)$$

(3.49)

with the only difference being the pulse shape $p_s(t) = \sin(\pi t / T)$ for $0 \leq t \leq T$ and zero otherwise. The sampling phase is an important factor in this case. We sample at times $t = mT_0 + \tau$, $T_0 = T / \nu$, where $\nu$ is the number of samples per symbol. For perfect sampling phase, $\tau = 0$. If the sampling phase is unknown, we model $\tau$ as uniform on $[0, T / \nu]$. The development of the ML estimate for $A$ is analogous to that in Case 1. We denote the random sampling instant as $\tau$ (shown in Figure 3.5). While performing interference cancellation, we need to cancel the strong signal component, $Ap_s(mT_0 + \tau)$. However, as this is the component which we are estimating, we apply the same procedure employed in our earlier case of rectangular pulses, in order to obtain $\tau$. The absolute value estimator is valid as the pulse shape is a constant “weighting” on $A$, for a given value of $m$. The explicit need for an estimate of $A$ would arise in multi-rate systems, where sampling rates differ in different parts of the system. In addition, explicit estimation of $A$ is of theoretical interest.
Analogous to (3.27), the pdf of the $i^{th}$ sample is

$$p(r_i \mid A, b_i, \tau) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp \left[ -\frac{\left( r_i - Ab_i \sin \left( \frac{\pi}{T_S} (kT_0 + \tau) \right) \right)^2}{2\sigma^2} \right]$$  \hspace{1cm} (3.50)$$

Since $b_i \in \{\pm 1\}$ equiprobably, we have

$$p(r_i \mid A) = \frac{1}{2\sqrt{2\pi\sigma^2}} \left[ e^{-\left( r_i - A \sin \left( \frac{\pi}{T_S} (kT_0 + \tau) \right) \right)^2/2\sigma^2} + e^{-\left( r_i + A \sin \left( \frac{\pi}{T_S} (kT_0 + \tau) \right) \right)^2/2\sigma^2} \right]$$  \hspace{1cm} (3.51)$$

which is analogous to (3.28) in the first case. When we expand the quadratics in the exponents, this becomes,

$$L(A) = \frac{1}{2\sqrt{2\pi\sigma^2}} \left[ e^{-\left| r_i^2 + p^2 \sin^2 \left( \frac{\pi}{T_S} (kT_0 + \tau) \right) \right|/2\sigma^2} + e^{-\left| r_i^2 + (kT_0 + \tau) + 2\pi p \sin \left( \frac{\pi}{T_S} (kT_0 + \tau) \right) \right|/2\sigma^2} \right]$$

where we the pulse shape function is
\[ p_s(kT_0 + \tau) = A \sin \left( \frac{\pi}{T_s}(kT_0 + \tau) \right) \]

Taking a natural logarithm of the likelihood function again, we obtain

\[ l(A) = \ln[L(A)] \]

\[ = \ln C + \sum_{i=1}^{N} \ln \left[ e^{-\left( r_i^2 + p_s^2(kT_0 + \tau) \right)/2\sigma^2} \left( e^{\left(2\text{log}_{2}(r_i^2)(kT_0 + \tau)\right)/2\sigma^2} + e^{-\left(2\text{log}_{2}(r_i^2)(kT_0 + \tau)\right)/2\sigma^2} \right) \right] \quad (3.52) \]

where \( C = \frac{1}{2\sigma \sqrt{2\pi}} \). Then (3.52) becomes

\[ l(A) = \ln C + \sum_{i=1}^{N} \left[ \frac{- \left( r_i^2 + p_s^2(kT_0 + \tau) \right)}{2\sigma^2} + \ln \left( 2 \cosh \left( r_i p_s(kT_0 + \tau) / \sigma^2 \right) \right) \right] \]

As before, the maximum likelihood estimate is obtained by taking the partial derivative of \( l(A) \) with respect to \( A \), and equating to zero. The derivative is

\[ \frac{\partial}{\partial A} \left[ l(A) \right] = \sum_{i=1}^{N} \frac{-2A \sin^2 \left( \frac{\pi}{T_s}(kT_0 + \tau) \right)}{2\sigma^2} + \frac{\partial}{\partial A} \ln \left[ 2 \cosh \left( r_i p_s(kT_0 + \tau) / \sigma^2 \right) \right] \quad \ldots (3.53) \]

The second term of this equation is,

\[ \frac{\partial}{\partial A} \ln \left[ 2 \cosh \left( r_i A \sin \left( \frac{\pi}{T_s}(kT_0 + \tau) \right) / \sigma^2 \right) \right] \]

\[ = \frac{1}{2 \cosh \left( r_i A \sin \left( \frac{\pi}{T_s}(kT_0 + \tau) \right) / \sigma^2 \right)} \frac{\partial}{\partial A} \left[ r_i A \sin \left( \frac{\pi}{T_s}(kT_0 + \tau) \right) / \sigma^2 \right] + 2 \cosh \left( r_i A \sin \left( \frac{\pi}{T_s}(kT_0 + \tau) \right) / \sigma^2 \right) \]

\[ = r_i \sin \left( \frac{\pi}{T_s}(kT_0 + \tau) \right) / \sigma^2 \tanh \left( \frac{r_i A \sin \left( \frac{\pi}{T_s}(kT_0 + \tau) \right)}{\sigma^2} \right) \left[ r_i A \sin \left( \frac{\pi}{T_s}(kT_0 + \tau) \right) / \sigma^2 \right] \]

\[ \quad \ldots (3.54) \]
Using (3.54) in (3.53), we have
\[
\frac{\partial}{\partial A} [l(A)] = -\frac{N A \sin^2 \left( \frac{\pi}{T_S} (iT_0 + \tau) \right)}{\sigma^2} + \sum_{i=1}^{N} \frac{r_i \sin \left( \frac{\pi}{T_S} (iT_0 + \tau) \right)}{\sigma^2} \tanh \left[ \frac{r_i A}{\sigma^2} \sin \left( \frac{\pi}{T_S} (iT_0 + \tau) \right) \right] = 0
\]
\[
\hat{A} = \frac{1}{N} \sum_{i=1}^{N} r_i \frac{\tanh \left[ \frac{r_i A}{\sigma^2} \sin \left( \frac{\pi}{T_S} (iT_0 + \tau) \right) \right]}{\sin \left( \frac{\pi}{T_S} (iT_0 + \tau) \right)} \tag{3.55}
\]
which is similar in form to (3.35) for the rectangular pulse case, except for weighting by the sinusoidal pulse function or its reciprocal. For our high SNR approximation, we obtain
\[
\hat{A} = \frac{1}{N} \sum_{i=1}^{N} r_i / \sin \left( \frac{\pi}{T_S} (iT_0 + \tau) \right) \tag{3.56}
\]
where we observe that the only difference between this case and (3.35) in Case 1 is the normalization by the sinusoidal pulse amplitude at the sample time.

For \(N_s\) samples/bit, we obtain \(p(r_{N_s,i} | A, \tau)\) analogous to (3.39) and via the exact same procedure as used in Case 1, we obtain
\[
\tilde{A} = \frac{1}{NN_s} \sum_{i=1}^{N} \sum_{m=0}^{N_s(i+1)-1} r_m \frac{\tanh \left( \frac{A}{\sigma^2} \sum_{m=0}^{N_s(i+1)-1} r_m p_s(mT_0 + \tau) \right)}{\sin \left[ \pi (mT_0 + \tau) / T \right]} \tag{3.57}
\]
and the high SNR case approximation is
\[
\tilde{A} = \frac{1}{NN_s} \sum_{i=1}^{N_s} \sum_{m=iN_i}^{(i+1)-1} r_m / \sin[\pi(mT_0 + \tau)/T]
\] (3.58)

This implies that we must know the delay \( \tau \) and normalize each of the \( N_s \) values accordingly. We assume in our numerical results that the delay \( \tau \) is known. In practice it could be estimated over the \( NN_s \) samples by averaging the \( N \) samples for each phase (e.g. \( r_0, r_{N_s}, r_{2N_s}, \ldots \) and \( r_1, r_{N_s+1}, r_{2N_s+1}, \ldots \)) to obtain a vector of \( N_s \) elements which represents an estimate of samples of \( \sin[\pi(mT_0 + \tau)/T] \). This estimate would be compared with a “template” of this function, over sampled at a high rate and stored in the receiver to obtain the estimate of delay \( \tau \).

### 3.2.4 Case 3: Rectangular Pulses, Two Users in AWGN

In our analysis here, we assume both signals are synchronized in carrier phase and symbol timing. This does not represent our case of interest, but is analytically tractable and will provide insight into our desired asynchronous case. The received sample is given by

\[
r_m = A_1 b_{1m} + A_2 b_{2m} + n_m
\] (3.59)

Here, \( A_1 \) is the amplitude of the signal to be estimated and \( A_2 \) is the amplitude of the interfering weak signal and \( b_m \in \{\pm 1\} \), where \( i \) is the user index. For this bit synchronous case, the pdf, analogous to (3.29), is given as

\[
p(r_i | A_1, A_2) = \int \int p(r | A_1, A_2, b) p(b_1) p(b_2) db_1 db_2
\] (3.60)
with \( b = (b_1, b_2) \). So any sample within \( r \) has pdf

\[
p(r_i | A_1, A_2) = \frac{1}{4\sqrt{2\pi\sigma^2}} \left\{ e^{-(r_i - A_1)^2/(2\sigma^2)} + e^{-(r_i - A_2)^2/(2\sigma^2)} + e^{-(r_i + A_1)^2/(2\sigma^2)} + e^{-(r_i + A_2)^2/(2\sigma^2)} \right\}
\]

or, we can express the joint pdf as

\[
p(r_{N_s} | A_1, A_2, b_{1,N_s}, b_{2,N_s}) = \prod_{m=1}^{N_s} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-(r_{m1} - A_1b_1 - A_2b_2)^2/(2\sigma^2)}
\]

(3.61)

\[
p(r | A_1, A_2) = \prod_{k=1}^{N} p(r_{N_s,k} | A_1, A_2)
\]

(3.62)

where \( N \) is the total number of bits and \( r_{N_s,k} \) contains all \( N_s \) samples of user 1’s \( k^{th} \) bit and all \( N_s \) samples of user 2’s \( k^{th} \) bit (for this synchronous case).

Since \( b_1, b_2 \in \{\pm 1\} \), (3.62) can be expressed as

\[
p(r_{N_s} | A_1, A_2) = \frac{1}{4\sqrt{2\pi\sigma^2}} \prod_{m=1}^{N_s} e^{-(r_{m1} - A_1)^2/(2\sigma^2)} + \frac{1}{4\sqrt{2\pi\sigma^2}} \prod_{m=1}^{N_s} e^{-(r_{m2} - A_2)^2/(2\sigma^2)}
\]

\[
+ \frac{1}{4\sqrt{2\pi\sigma^2}} \prod_{m=1}^{N_s} e^{-(r_{m1} + A_1)^2/(2\sigma^2)} + \frac{1}{4\sqrt{2\pi\sigma^2}} \prod_{m=1}^{N_s} e^{-(r_{m2} + A_2)^2/(2\sigma^2)}
\]

or

\[
p(r_{N_s} | A_1, A_2) = \frac{1}{4\sqrt{2\pi\sigma^2}} \sum_{m=1}^{N_s} e^{-\sum_{j=1}^{N} (r_{mj} - c_j)^2/(2\sigma^2)}
\]

(3.63a)

where \( c_j, j = 1,2,3,4 \) is defined as
Using (3.63b) in (3.63a), the pdfs for i.i.d. data bit vectors, we obtain

\[
p(r \mid A_1, A_2) = C_N \prod_{k=1}^{N} \sum_{t=1}^{4} e^{-\frac{(b_{1k} - c_{1t})^2}{2\sigma^2}} \prod_{m=1}^{kN_s} e^{-\frac{(b_{1m} - c_{1t})^2}{2\sigma^2}}
\]

where \( C_N = [4\sqrt{2\pi\sigma^2}]^{-N} \).

We observe that the log of the partial derivative of the likelihood function cannot be simplified further without making further assumptions. If the signal to noise ratio is large enough for the strong signal \( A_i \gg \sigma \) and if the near far ratio \( A_1/A_2 \) is also large, the derivative of the log-likelihood function is approximately given by

\[
\frac{\partial \ln[L(A_1, A_2)]}{\partial A} \approx -NN_sA_i + \sum_{m=(k-1)N_s+1}^{kN_s} r_m \text{sgn}(r_m)
\]

and if we set this to zero, we obtain a result identical to (3.40) for the single user case (as we should).

We can obtain the pdf for the case when we have phase synchronism and timing (symbol) asynchronism, but we still cannot obtain the ML estimate in any useful closed form. Hence we resort to simulations to explore this and the most realistic case--both
symbol and phase asynchronism. We expect intuitively that the symbol and phase synchronous case is the worst case from an estimation perspective. This is because when symbols are aligned, the weaker signal can distort the received signal by the largest amount. These intuitive observations will be shown to hold in all our simulation results.

3.2.5. Case 4: Sinusoidal Pulses, Two Users in AWGN

The received baseband signal is the sum of two signals analogous to (3.37) plus Gaussian noise. As with the rectangular pulse case, we consider the symbol and carrier phase synchronous conditions for analytical tractability.

The pdf analogous to (3.59) is given as

$$p(r_{Ns} | A_1, A_2, b_{1,Ns}, b_{2,Ns}) = \prod_{m=1}^{N_s} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\left((r_m-A_1p_{1}(t-iT)-A_2p_{2}(t-iT))^2/(2\sigma^2)\right)}$$

where $p_s(t)$ is the sinusoidal pulse defined after (3.49)

The pdf of the received sample vector, analogous to (3.65) is

$$p(r | A_1, A_2) = C_N \prod_{k=1}^{N} \sum_{i=1}^{4} e^{-\sum_{\alpha=\delta}^{(k+l)N_s-1} \left[(r_{\alpha-c_{j,p_{c_{m,T_0+r}}})^2]}\right]}$$

where the set of $c_j$'s is defined as in the previous section. As before, if we assume both high SNR and high near-far ratio, we get the approximate result of (3.58), but cannot proceed further without approximations. Similarly, we can obtain the pdf for the case when we have phase asynchronism, but an accurate expression for the estimate of the amplitude cannot be obtained. We make use of simulations to validate these intuitions.
Chapter 4

Numerical Results

In this section, we provide simulation results for the four cases discussed earlier and for various parameter values. As in [6], we utilize the normalized mean square error (NMSE) as the performance criterion. It is defined as

\[
NMSE = \left( \frac{A - \hat{A}}{A} \right)^2
\]

(4.1)

where \( \hat{A} \) is the estimate of the amplitude and \( A \) is the true amplitude. As discussed in the previous chapter, we consider two settings: a single user in AWGN, and two users in AWGN. In the two user sinusoidal pulse shape setting, we explore four possible combinations of bit and phase synchronism:

- bit synchronous, phase synchronous
- bit synchronous, phase asynchronous
- bit asynchronous, phase synchronous
- bit asynchronous, phase asynchronous.

We also used the common root-mean-square estimator for comparative studies. The simulations were conducted in MATLAB®, and averaged over 200 trials for each point of all curves. The system model is as shown in Figure 2.3.1
4.1. Single User in AWGN

In Figure 4.1, we plot the true amplitude and its estimates vs. block size $N$ for different values of SNR ($E_b/N_0$) for rectangular pulses. Results for two kinds of estimators are shown: the absolute value estimator, defined as follows

$$\hat{A}_{\text{abs}} = \frac{1}{N} \sum_{i=1}^{N} |r_i|,$$  \hspace{1cm} (4.2)

and the root mean square (rms) estimator, expressed as

$$\hat{A}_{\text{rms}} = \sqrt{\frac{1}{N} \sum_{i=1}^{N} (r_i)^2}.$$  \hspace{1cm} (4.3)

It can be observed that as $N$ increases, the performance of the estimators improves (they gradually approach the true value of the amplitude). From Figure 4.1, it is clear that the absolute value estimate is more accurate than the rms estimate. This is due to the fact that the amplitude of the signal (with noise included), when squared causes a larger shift away from the true value of amplitude, when compared to the absolute value of the signal; that is, the rms estimator bias is larger.
Figure 4.1. Plot of amplitude vs. block size $N$ for different values of $E_b/N_0$ (3, 6 and 10 dB). True amplitude, absolute value, and rms estimate shown.

In Figure 4.2, we plot the NMSE vs. estimator block size $N$ for several values of SNR for rectangular pulses. Comparative results for the cases of more than one sample per symbol ($N_s > 1$) and a single sample per symbol, $N_s = 1$, for different values of SNR, are shown. As expected, oversampling provides slightly faster convergence at the expense of faster processing. It is also observed that the NMSE decreases as the SNR increases. The NMSE decreases as the block size $N$ increases, also as expected.
Figure 4.2. NMSE vs. block size N, single user, rectangular pulses for Ns=1 and 2 samples/symbol, and different values of Eb/N0: 3, 6 and 10 dB.

In Figure 4.3, we show NMSE vs. $E_b/N_0$ for a fixed value of $N=50$, for three different estimators: the full ML estimator of (3.30) which uses knowledge of the true amplitude; the absolute value estimator approximation to (4.2); and the rms estimator, (4.3). As observed, the absolute value approximation to the $tanh$ function is very accurate for $E_b/N_0$ as low as 3 dB as it is essentially indistinguishable from the true ML estimator for SNRs above this value.
Figure 4.3. NMSE vs. Eb/N0, single user, rectangular pulses for N=50. Full MLSE, absolute value, and rms estimates.

Figure 4.4 shows results similar to those of Figure 4.2, again for a single user in AWGN, but for the case of sinusoidal pulses. It shows results for both the absolute value estimator and the rms estimator. The larger bias due to noise in the rms estimator is clearly evident. The trials were run for 3 different values of SNR. As expected, for smaller SNR values, the bias is comparatively larger.
Figure 4.4. NMSE vs. N, single user, sinusoidal pulses, several values of $E_b/N_0$: 3, 6 and 10dB. ‘rms’ estimate also shown

4.2. Two Users in AWGN

In this case, apart from showing the variation of NMSE vs. block length and SNR, we also show the effects of near-far ratio (NFR), phase asynchronism, and bit asynchronism. In Figure 4.5, we have shown plots of NMSE vs. N, for several values of $E_b/N_0$ of the stronger signal and NFR and two conditions. The two conditions shown are both bit and phase synchronous, and both bit and phase asynchronous. As expected,
when the interfering user is both bit and phase synchronous with the stronger signal, estimating the amplitude of the stronger signal is inherently less accurate, hence yielding a higher NMSE. The variation of NMSE as a function of $E_b/N_0$ is as expected—as $E_b/N_0$ increases, the NMSE decreases. Likewise, as a function of NFR, the NMSE decreases as the NFR increases as expected. The plot shows possible combinations of 2 values each for both $E_b/N_0$ and NFR. As expected, the maximum error is obtained when the $E_b/N_0$ and NFR are smallest.

![Figure 4.5. NMSE vs. N, rectangular pulses, two users and two conditions: bit and phase synchronous, and bit and phase asynchronous, for different values of Eb/N0 and NFR.](image-url)
In Figure 4.6, we have shown plots of NMSE vs. $N$ for the sinusoidal pulse shape two user case. For all the curves in this figure, the two user signals are bit synchronous. The effect of phase synchronism on performance is illustrated for three different values of $E_b/N_0$ and NFR. As anticipated, the performance is better (NMSE is lower) when the carrier phase is not aligned, as estimation of amplitude is less accurate when there is phase alignment. This can be explained as follows: when the two signals exhibit phase synchronism, identifying the stronger signal uniquely more difficult, because each quadrature component data symbol is aligned in time, hence the weaker signal can distort the stronger signal by the largest amount.

Figure 4.6. NMSE vs. $N$, sinusoidal pulses, two users and two conditions—bit and phase synchronous, and bit synchronous and phase asynchronous, for different values of $E_b/N_0$ and NFR
When the two signals are phase asynchronous, there is inter-channel interference (I to Q, and vice-versa), which tends to randomize the interference the weak signal presents to the stronger signal we are estimating.

Figure 4.7 illustrates the remaining two conditions for the sinusoidal pulse, two user case: bit asynchronous reception with both phase synchronism and asynchronism. Again as expected, we observe the best performance in the most practical case when both bit and carrier phases are asynchronous between the two signals.

Fig 4.7. NMSE vs. N, sinusoidal pulses, two users and two conditions--bit asynchronous and phase synchronous, and bit and phase asynchronous, for different values of $E_b/N_0$ and NFR.
In Figure 4.8, we plot the bit error probability (BER) for the weak signal vs. NFR for different values of SNR with the use of a single stage of interference canceling, as described in section 2.3.1. The interference cancellation is performed assuming knowledge of all $N$ bits for the strong user, perfect timing for the strong user and perfect carrier phase synchronism. Performance with both the true value of amplitude and the amplitude estimated over the $N$ bits using our absolute value estimator is shown. The BER decreases as the NFR increases as the probability of error in the estimate of amplitude of the stronger signal decreases as the ratio of the stronger to the weaker signal increases, but the BER reaches a natural “floor” at larger values NFR. This “floor” represents the case of the strong signal alone, in the presence of AWGN. The plot obtained using the estimate of the amplitude of the stronger user has a slightly higher bit error rate when compared to the plot obtained using the exact value of the amplitude of the stronger signal, as shown below. This minute difference is because the amplitude cannot be estimated “perfectly” (100% accuracy) using the estimator, giving rise to a small error.
Figure 4.8. Plot of NMSE vs. NFR for users (phase and bit asynchronous) for different values of Eb/N0: 2, 4, and 6 dB. Plot using exact values of amplitude of strong signal also shown.
Chapter 5

Summary, Conclusions and Future Work

In this chapter, we summarize the research performed in this thesis. We provide an overview of the results and conclusions. We also discuss possible future work.

5.1 Summary and Conclusions

In this thesis, we explored the use of maximum likelihood amplitude estimation for MSK in co-channel interference. The absolute value estimator was found to be more accurate than the root mean square value estimate in terms of the performance measure of normalized mean-square error. To connect with previous work and provide a logical course of development, we considered four different cases, consisting of one and two users, each with either rectangular or sinusoidal pulses. For the single-user cases, ML amplitude estimates were analytically derived and when the SNR is moderately high, are well approximated by simple absolute value estimators. In the case of two users, the exact ML amplitude estimate cannot be derived in closed form; for certain conditions (e.g. high SNR and high NFR), it can be approximated by the single user and/or absolute value estimator. We also determined values for the absolute value estimator bias, and showed that it varies inversely (as expected) as the square root of the SNR of the strong signal but has no dependence upon sample size $N$. The
variance for the estimator was analyzed was confirmed to be consistent. The variance however, was found to be dependent on the sample size $N$.

We conducted computer simulations to explore the performance of the absolute value estimators for MSK in CCI with different conditions, including various values of NFR, estimator block length, number of samples per symbol, and the presence or absence of both symbol and phase synchronism. It was shown that the absolute value estimator performs very well in terms of NMSE across a broad range of conditions. As expected, its performance for the most practical case of interest--phase and symbol asynchronism with disparate signal strengths--was exceptional.

5.2 Future Work

In order to effectively suppress the interference, we need to obtain an accurate estimate of the Doppler shift in frequency in addition to the amplitude estimate. In Chapter 2, we discussed how Doppler estimation could be performed independently (and prior to) amplitude estimation. One item of future work would hence be to find a joint estimator for amplitude and Doppler shift. Another area for future work would be to develop additional amplitude estimators. The existing amplitude estimator could also be combined with decision direction to obtain a more accurate estimate of the amplitude. Using decision direction would help reduce the bias, shown in Figure 4.8. Finally, more analytical work on quantifying estimator performance should be done.
References:


Appendix: Matlab Programs

% Amplitude Estimation of MSK in Co-Channel Interference
% In this program we plot the Normalized mean square error (NMSE) against the block size (N) for 3 different values of signal to noise ratio for a rectangular pulse stream
% in the presence of additive white gaussian noise (AWGN)
%
% Ravikanth Ekanthalingam
% Ohio University, Athens.
%
% This program assumes Eb/N01=6 dB and NFR=6 dB phase asynchronous

clc;
close all;
clear all;
j=sqrt(-1);
BER1=0;              % If user-1 BER computation desired, set BER1=1; **NOT QUITE ACCURATE IN SECTION AT BOTTOM***
fsT=4;k=1;     % Normalized sampling rate, per bit (fs=sampling frequency, T=bit time=1 here)
T=1;Ts=2*T;          % Set bit and symbol durations
Ns=2*fsT;            % Ns=#samples/symbol=#samples/Ts=2*fsT
Ntrials=500;           % # trials over which to average amp estimate quality, for a given Eb1/N0, N, and NFS
Eb1=6;       % Energy per bit to spectral noise density ratio (Eb/N0) for user 1, in dB
Eb1n=10^(Eb1/10);    % Eb/N0 for user 1, numeric; (user 1 is "strong" signal)
A1=sqrt(2*Eb1n/fsT); % Amplitude of user 1 MSK signal (T=1)
NFR=6;       % Near-far ratio, in dB (Eb/N0)strong/(Eb/N0)weak
A2=A1*(10^(-NFR/20));% Amplitude of user 2 MSK signal
Eb2n=(A2^2)*fsT/2;   % Eb/N0 for user 2, numeric; (user 2 is "weak" signal)
Eb2=10*log10(Eb2n);  % Eb/N0 for user 2, in dB
for N=10:5:100 ; % Number of BITS used, must be EVEN (N/2 symbols on both I and Q) if mod(N,2) ~= 0; N=N+1; end

t=0:1/fsT:N-1/fsT;% Create time vector for plots
Lv=fsT*N; % Length of complex transmitted vectors v1 and v2
Lv=Lv+fsT; % Extend vector by one bit, fsT samples
NMSE=0; % Initialize the normalized mean-square amplitude estimation error
NMSErs=0; % Initialize normalized MS amp est error for simple sqrt(rI^2+rQ^2) estimator
for ik=1:Ntrials
    fD2=0.000001; % Doppler for co-channel signal (signal 2); set VERY low
    phi2=2*pi*rand(1,1); % Random phase for co-channel signal
    taus=rand(1,1)*1/fsT; % Sampling phase offset for strong signal 1, uniform between 0 and
    Ts/Ns=Ts/2/fsT=2*T/2/fsT=1/fsT
taus=0; % Set sampling phase offset to zero for testing
    % For burst Asynchronous reception, lengthen vectors; otherwise, just circularly shift signal 2
    LvA=3*Lv; % Use the following for burst Async
    % User 1 signal transmitted as [zeros(1,Lv) v1 zeros(1,Lv)], i.e., length-Lv vector centered between
    % two length-Lv zero vectors
    % User 2 signal transmitted as [zeros(1,tau) v2 zeros(1,2*Lv-tau)], i.e., length-Lv vector at arbitrary
    % delay tau
    % Generate MSK via OFFSET QPSK, filtered with a sinusoidal pulse.
    u=sin(pi/Ts*(0:1/fsT:Ts-1/fsT)+taus*pi/Ts); % Create sinusoidal pulse shape
    Nover=32;
    uV=sin(pi/2*(0:1/(Nover/2):2-1/(Nover/2))); % Create "template" pulse, Nover samples/Ts, to
    % which estimate can be compared
    b1=BIN01(N,0.5); b2=BIN01(N,0.5); % Generate random binary vectors
    bover1=OverN(b1,fsT); bover2=OverN(b2,fsT); % Oversample
    I1=zeros(1,N*fsT+fsT); Q1=zeros(1,N*fsT+fsT); % Initialize I1 and Q1 components
    I2=zeros(1,N*fsT+fsT); Q2=zeros(1,N*fsT+fsT); % Initialize I2 and Q2 components
    for ii=1:2:N; % Loop to create MSK baseband vectors
        I1((ii-1)*fsT+1:(ii+1)*fsT) = (-2*b1(ii)+1)*u; % '0' maps to 1; '1' maps to -1
        Q1(ii*fsT+1:(ii+2)*fsT) = (-2*b1(ii+1)+1)*u;
        I2((ii-1)*fsT+1:(ii+1)*fsT) = (-2*b2(ii)+1)*u;
        Q2(ii*fsT+1:(ii+2)*fsT) = (-2*b2(ii+1)+1)*u;
    end
    % Generate Doppler envelope for CC interfering signal (signal 2)
    Ad2c=cos(2*pi*fD2*[0:N*fsT+fsT-1]/fsT+phi2);
Ad2s = sin(2*pi*fD2*[0:N*fsT+fsT-1]/fsT+phi2);

% Generate baseband MSK signals, with Doppler envelope (only on weak signal v2)
% If NO Doppler, vk=Ak*(I_k-j*Q_k), where k=1 or 2
v2 = A2*(I2(1:N*fsT+fsT).*Ad2c-Q2(1:N*fsT+fsT).*Ad2s);

% For burst Asynchronous, set bA=1
bA = 0;
if bA == 1
    v1 = [zeros(1,Lv) v1 zeros(1,Lv)];       % Lengthen signal 1 to 3*Lv for async
    tau = floor(rand(1,1)*(2*Lv+1));       % User 2 random delay, between 0 and 2*Lv samples
    v2 = [zeros(1,tau) v2 zeros(1,2*Lv-tau)];     % User 2 signal, async
    n1 = (randn(1,LvA)+j*randn(1,LvA))*1;         % Baseband AWGN
    rA1 = v1 + v2 + n1;                           % Baseband received signal
    rs1 = rA1(Lv+1:2*Lv);                        % Extract signal 1 subsequence from r
else
    v2 = cshift(v2, Randinteg(1,1,fsT-1));
    n1 = randn(1,Lv)+j*randn(1,Lv);
    rA1 = v1 + v2 + n1;
    rs1 = rA1;                                    % Extract signal 1 subsequence from r
end

rAe = sqrt(sum(real(rs1).^2+imag(rs1).^2)/length(rs1));   % sqrt(rI^2 + rQ^2 ) amp estimate
NMSErs = NMSErs+(1-rAe/A1)^2/Ntrials;

% Now, amplitude estimation of user 1 signal............
I1hat = real(rs1(1:N*fsT));  Q1hat = imag(rs1(1:N*fsT+fsT));

% Organize samples by position within each symbol--get 2*fsT samples/symbol, so 2*fsT sub-vectors
Imat = reshape(I1hat, 2*fsT,N/2); Qmat = reshape(Q1hat, 2*fsT,N/2);

Im = Imat'; Qm = Qmat';  % Each column of Im (or Qm) has samples in same position

% uhatl and uhatQ are each vectors of 2*fsT samples, estimate of pulse sin(pi*(t+taus)/(2*T)),
t = k*fsT, k=0,1,...2*fsT-1

uhatl = 2*sum(abs(Im))/N;  uhatQ = 2*sum(abs(Qm))/N;

% Next need to "extract" amplitude A1 from these averaged pulse samples;
uest = (uhatl+uhatQ)/2;     % Averaging of BOTH I and Q samples together
uestS = overN(uest/A1,Nover/Ns);       % Oversample estimate for determining delay estimate tauhat
(assumes "perfect" scaling with true A1)
uev = abs(uestS-uV);         % Compare oversampled estimate with "template" pulse uV
[du,iu] = min(uev); % Find index of minimum difference between estimate & template
mdelay = mod(iu, Nover/Ns); % NOTE: NOT CERTAIN OF ACCURACY of the "if" function here
if mdelay == 0;
    mdelay;
    tauh = ((Nover/Ns) - 1)/Nover;
else
    tauh = (mod(iu, Nover/Ns) - 1)/Nover; % tauh is quantized version of tauhat, in increments of 1/Nover
end
tauhat = Ts*tauh; % Final estimate of taus
u2 = sin(pi/Ts*(0:1/fsT:Ts-1/fsT)+tauhat*pi/Ts);
A1hat = max(uest)/max(u2);
NMSE = NMSE + (1 - A1hat/A1)^2/Ntrials;
end
p(k) = NMSE
k = k + 1;
end
%
q(1) = 10; q(2) = 15; q(4) = 25; q(6) = 35; q(8) = 45; q(10) = 55; q(12) = 65; q(14) = 75; q(16) = 85; q(18) = 95; q(3) = 20; q(5) = 30; q(7) = 40; q(9) = 50; q(11) = 60; q(13) = 70; q(15) = 80; q(17) = 90; q(19) = 100;
% plot(q, p);
% grid on;
% hold on;
% xlabel('N -->');
% ylabel('NMSE -->');
% title('NMSE vs N for E_b_1/N_o=6 dB and NFR=3 dB');
% legend('phase asynchronous');
% SECTION BELOW from MSK3, for computing Pb of user 1
if BER1 == 1; % Compute user 1 Pb estimate if flag set
    Pb1 = 0; Pb2 = 0; % Initialize Pb's
    Ih11 = zeros(1, N*fsT); Qh11 = zeros(1, N*fsT+fsT); % Initialize I1 and Q1 component estimates
    Ih12 = zeros(1, N*fsT); Qh12 = zeros(1, N*fsT+fsT); % Initialize I2 and Q2 component estimates
    % Next, receiver correlation and detection for signal 1, assuming estimated carrier freq & phase, & perfect symbol timing
    for ir = 1:2:N
        Ih11((ir-1)*fsT+1:(ir+1)*fsT) = u.*real(rs1((ir-1)*fsT+1:(ir+1)*fsT))*sqrt(2);
Ihat11((ir+1)/2)=sign(sum(Ih11((ir-1)*fsT+1:(ir+1)*fsT))); %I-channel estimate of signal 1, hard decision
Ihat11((ir+1)/2)=sum(Ih11((ir-1)*fsT+1:(ir+1)*fsT)); %I-channel estimate of signal 1, soft decision
Qh11(ir*fsT+1:(ir+2)*fsT)=-u.*imag(rs1(ir*fsT+1:(ir+2)*fsT))*sqrt(2);
Qhat11((ir+1)/2)=sign(sum(Qh11(ir*fsT+1:(ir+2)*fsT)));%Q-channel estimate of signal 1, hard decision
Qhat11((ir+1)/2)=sum(Qh11(ir*fsT+1:(ir+2)*fsT));% Q-channel estimate of signal 1, soft decision
end
bh11=LEAVE2(Ihat11,Qhat11); % Interleave soft I and Q estimates for signal 1
bh1=(1-sign(bh11))/2; % Hard decision estimates for signal 1
bh1H=sign(bh1); % Hard bit decisions on signal 1
bhat1=(1-bh1H)/2; % Convert +/-1 values to 0/1 values
er1=abs(b1-bhat1); % Count bit errors, and compute Pb1 estimate
Pb1=sum(er1)/N;
end

%This part assumes Eb/N01=10 dB and NFR=3 dB phase asynchronous
j=sqrt(-1);
BER1=0; % If user-1 BER computation desired, set BER1=1; **NOT QUITE ACCURATE IN SECTION AT BOTTOM***
fsT=4;k=1; % Normalized sampling rate, per bit (fs=sampling frequency,T=bit time=1 here)
T=1;Ts=2*T; % Set bit and symbol durations
Ns=2*fsT; % Ns=#samples/symbol=#samples/Ts=2*fsT
Ntrials=500; % # trials over which to average amp estimate quality, for a given Eb1/N0, N, and NFS
Eb1=10; % Energy per bit to spectral noise density ratio (Eb/N0) for user 1, in dB
Eb1n=10^(Eb1/10); % Eb/N0 for user 1, numeric; (user 1 is "strong" signal)
A1=sqrt(2*Eb1n/fsT); % Amplitude of user 1 MSK signal (T=1)
NFR=3; % Near-far ratio, in dB (Eb/N0)strong/(Eb/N0)weak
A2=A1*(10^(-NFR/20));% Amplitude of user 2 MSK signal
Eb2n=(A2^2)*fsT/2; % Eb/N0 for user 2, numeric; (user 2 is "weak" signal)
Eb2=10*log10(Eb2n); % Eb/N0 for user 2, in dB
for N=10:5:100 ; % Number of BITS used, must be EVEN (N/2 symbols on both I and Q)
    if mod(N,2) ~= 0; N=N+1; end
    t=0:1/fsT:N-1/fsT;% Create time vector for plots
\( \text{Lv}=\text{fsT*\text{N}}; \quad \% \text{Length of complex transmitted vectors } v1 \text{ and } v2 \)
\( \text{Lv}=\text{Lv}+\text{fsT}; \quad \% \text{Extend vector by one bit, } \text{fsT} \text{ samples} \)
\( \text{NMSE}=0; \quad \% \text{Initialize the normalized mean-square amplitude estimation error} \)
\( \text{NMSErs}=0; \quad \% \text{Initialize normalized MS amp est error for simple } \sqrt{rI^2+rQ^2} \text{ estimator} \)
\begin{verbatim}
for ik=1:Ntrials
  fD2=0.000001; \quad \% \text{Doppler for co-channel signal (signal 2); set VERY low}
  phi2=0; \% \text{Random phase for co-channel signal}
  taus=rand(1,1)*1/fsT; \% \text{Sampling phase offset for strong signal 1, uniform between 0 and}
  Ts/Ns=Ts/2/fsT=2*T/2/fsT=1/fsT
  taus=0; \% \text{Set sampling phase offset to zero for testing}
  \% For burst Asynchronous reception, lengthen vectors; otherwise, just circularly shift signal 2
  \text{LvA}=3*\text{Lv}; \quad \% \text{Use the following for burst Async}
  \% User 1 signal transmitted as \([\text{zeros}(1,\text{Lv}) \ v1 \ \text{zeros}(1,\text{Lv})]\), i.e., length-Lv vector centered between
  \% two length-Lv zero vectors
  \% User 2 signal transmitted as \([\text{zeros}(1,\text{tau}) \ v2 \ \text{zeros}(1,2*\text{Lv}-\text{tau})]\), i.e., length-Lv vector at arbitrary
  \% delay tau
  \% Generate MSK via OFFSET QPSK, filtered with a sinusoidal pulse.
  u=sin(pi/Ts*(0:1/fsT:Ts-1/fsT)+taus*pi/Ts); \% Create sinusoidal pulse shape
  Nover=32;
  uV=sin(pi/2*(0:1/(Nover/2):2-1/(Nover/2))); \% Create "template" pulse, Nover samples/Ts, to
  \% estimate can be compared
  b1=BIN01(N,0.5); \quad b2=BIN01(N,0.5); \% Generate random binary vectors
  bover1=OverN(b1,fsT); \quad bover2=OverN(b2,fsT); \% Oversample
  I1=zeros(1,N*fsT+fsT); \quad Q1=zeros(1,N*fsT+fsT); \% Initialize I1 and Q1 components
  I2=zeros(1,N*fsT+fsT); \quad Q2=zeros(1,N*fsT+fsT); \% Initialize I2 and Q2 components
  for ii=1:2:N;
    I1((ii-1)*fsT+1:(ii+1)*fsT) = (-2*b1(ii)+1)*u; \% '0' maps to 1; '1' maps to -1
    Q1((ii+1)*fsT+1:(ii+2)*fsT) = (-2*b1(ii+1)+1)*u;
    I2((ii-1)*fsT+1:(ii+1)*fsT) = (-2*b2(ii)+1)*u;
    Q2((ii+2)*fsT+1:(ii+1)*fsT) = (-2*b2(ii+1)+1)*u;
  end
  \% Generate Doppler envelope for CC interfering signal (signal 2)
  \text{Ad2c=cos(2*pi*fD2*[0:N*fsT+fsT-1]/fsT+phi2);}
  \text{Ad2s=sin(2*pi*fD2*[0:N*fsT+fsT-1]/fsT+phi2);}
  \% Generate baseband MSK signals, with Doppler envelope (only on weak signal v2)
  \% If NO Doppler, \( v_k=A_k*(I_k-j*Q_k) \), where \( k=1 \) or 2
\end{verbatim}
v2=A2*(I2(1:N*fsT+fsT).*Ad2c-Q2(1:N*fsT+fsT).*Ad2s);
v2=v2+j*A2*(-I2(1:N*fsT+fsT).*Ad2s+Q2(1:N*fsT+fsT).*Ad2c);
v1=A1*(I1(1:N*fsT+fsT)-j*Q1(1:N*fsT+fsT));

% For burst Asynchronous, set bA=1
bA=0;
if bA == 1
    v1=[zeros(1,Lv) v1 zeros(1,Lv)]; % Lengthen signal 1 to 3*Lv for async
    tau=floor(rand(1,1)*(2*Lv+1)); % User 2 random delay, between 0 and 2*Lv samples
    v2=[zeros(1,tau) v2 zeros(1,2*Lv-tau)]; % User 2 signal, async
    n1=(randn(1,LvA)+j*randn(1,LvA))*1; % Baseband AWGN
    rA1=v1+v2+n1; % Baseband received signal
    rs1=rA1(Lv+1:2*Lv); % Extract signal 1 subsequence from r
else
    v2=cshift(v2,Randinteg(1,1,fsT-1));
    n1=randn(1,Lv)+j*randn(1,Lv);
    rA1=v1+v2+n1;
    rs1=rA1; % Extract signal 1 subsequence from r
end

rAe=sqrt(sum(real(rs1).^2+imag(rs1).^2)/length(rs1)); % sqrt(rI^2 + rQ^2 ) amp estimate
NMSErs=NMSErs+(1-rAe/A1)^2/Ntrials;

% Now, amplitude estimation of user 1 signal................
I1hat=real(rs1(1:N*fsT)); Q1hat=-imag(rs1(fsT+1:N*fsT+fsT));

% Organize samples by position within each symbol--get 2*fsT samples/symbol, so 2*fsT sub-vectors
Imat=reshape(I1hat,2*fsT,N/2); Qmat=reshape(Q1hat,2*fsT,N/2);

Im=Imat'; Qm=Qmat' ; % Each column of Im (or Qm) has samples in same position
% uhatI and uhatQ are each vectors of 2*fsT samples, estimate of pulse sin(pi*(t+taus)/(2*T)),
t=k*fsT, k=0,1,...2*fsT-1
uhatI=2*sum(abs(Im))/N; uhatQ=2*sum(abs(Qm))/N;
% Next need to "extract" amplitude A1 from these averaged pulse samples;
uest=(uhatI+uhatQ)/2; % Averaging of BOTH I and Q samples together
uestS=overN(uest/A1,Nover/Ns); % Oversample estimate for determining delay estimate tauhat

(assumes "perfect" scaling with true A1)

uev=abs(uestS-uV); % Compare oversampled estimate with"template" pulse uV
[du,iu]=min(uev); % Find index of minimum difference between estimate & template
mdelay=mod(iu,Nover/Ns); % NOTE: NOT CERTAIN OF ACCURACY of the "if" function here
if mdelay == 0;
mdelay;

tauh=\frac{(Nover/Ns)-1}{Nover};

\text{else} \quad \text{tauh=mod(iu,Nover/Ns)-1/Nover;} \quad \% \text{tauh is quantized version of tauhat, in increments of 1/Nover}
\text{end}

tauhat=Ts*tauh; \quad \% \text{Final estimate of taus}

u2=sin(pi/Ts*(0:1/fsT:Ts-1/fsT)+tauhat*pi/Ts);
A1hat=max(uest)/max(u2);
NMSE=NMSE+(1-A1hat/A1)^2/Ntrials;
\text{end}
p1(k)=NMSE
k=k+1;
\text{end}

\% \text{SECTION BELOW from MSK3, for computing Pb of user 1}
\text{if BER1 == 1;} \quad \% \text{Compute user 1 Pb estimate if flag set}
Pb1=0; \quad Pb2=0; \quad \% \text{Initialize Pb's}
Ih11=zeros(1,N*fsT); \quad Qh11=zeros(1,N*fsT+fsT); \quad \% \text{Initialize I1 and Q1 component estimates}
Ih12=zeros(1,N*fsT); \quad Qh12=zeros(1,N*fsT+fsT); \quad \% \text{Initialize I2 and Q2 component estimates}
\% \text{Next, receiver correlation and detection for signal 1, assuming estimated carrier freq & phase, & perfect symbol timing}
\text{for ir=1:2:N}
  Ih11((ir-1)*fsT+1:(ir+1)*fsT)=u.*real(rs1((ir-1)*fsT+1:(ir+1)*fsT))*sqrt(2);
  Ihat11((ir+1)/2)=sign(sum(Ih11((ir-1)*fsT+1:(ir+1)*fsT))); \quad \% \text{I-channel estimate of signal 1, hard decision}
  Ihat11((ir+1)/2)=sum(Ih11((ir-1)*fsT+1:(ir+1)*fsT)); \quad \% \text{I-channel estimate of signal 1, soft decision}
  Qh11(ir*fsT+1:(ir+2)*fsT)=-u.*imag(rs1(ir*fsT+1:(ir+2)*fsT))*sqrt(2);
  Qhat11((ir+1)/2)=sign(sum(Qh11(ir*fsT+1:(ir+2)*fsT))); \quad \% \text{Q-channel estimate of signal 1, hard decision}
  Qhat11((ir+1)/2)=sum(Qh11(ir*fsT+1:(ir+2)*fsT)); \quad \% \text{Q-channel estimate of signal 1, soft decision}
\text{end}
bh11=LEAVE2(Ihat11,Qhat11); \quad \% \text{Interleave soft I and Q estimates for signal 1}
bh1=(1-sign(bh11))/2; \quad \% \text{Hard decision estimates for signal 1}
bh1H=sign(bh1); \quad \% \text{Hard bit decisions on signal 1}
bhat1=(1-bh1H)/2; \quad \% \text{Convert +/-1 values to 0/1 values}
er1=abs(b1-bhat1); \quad \% \text{Count bit errors, and compute Pb1 estimate}
Pb1=sum(er1)/N;
% This part assumes Eb/N01=10 dB and NFR=6 dB phase asynchronous

j=sqrt(-1);
BER1=0;  % If user-1 BER computation desired, set BER1=1; **NOT QUITE ACCURATE IN SECTION AT BOTTOM***
fsT=4;k=1;     % Normalized sampling rate, per bit (fs=sampling frequency, T=bit time=1 here)
T=1;Ts=2*T;          % Set bit and symbol durations
Ns=2*fsT;            % Ns=#samples/symbol=#samples/Ts=2*fsT
Ntrials=500;           % # trials over which to average amp estimate quality, for a given Eb1/N0, N, and NFS
Eb1=10;               % Energy per bit to spectral noise density ratio (Eb/N0) for user 1, in dB
Eb1n=10^(Eb1/10);    % Eb/N0 for user 1, numeric; (user 1 is "strong" signal)
A1=sqrt(2*Eb1n/fsT); % Amplitude of user 1 MSK signal (T=1)
NFR=6;               % Near-far ratio, in dB (Eb/N0)strong/(Eb/N0)weak
A2=A1*(10^(-NFR/20));% Amplitude of user 2 MSK signal
Eb2n=(A2^2)*fsT/2;   % Eb/N0 for user 2, numeric; (user 2 is "weak" signal)
Eb2=10*log10(Eb2n);  % Eb/N0 for user 2, in dB
for N=10:5:100            ;    % Number of BITS used, must be EVEN (N/2 symbols on both I and Q)
  if mod(N,2) ~= 0; N=N+1; end
  t=0:1/fsT:N-1/fsT;  % Create time vector for plots
  Lv=fsT*N;       % Length of complex transmitted vectors v1 and v2
  Lvb=Lv+fsT;     % Extend vector by one bit, fsT samples
  NMSE=0;         % Initialize the normalized mean-square amplitude estimation error
  NMSErs=0;       % Initialize normalized MS amp est error for simple sqrt(rI^2+rQ^2) estimator
  % For burst Asynchronous reception, lengthen vectors; otherwise, just circularly shift signal 2
  LvA=3*Lv;        % Use the following for burst Async
% User 1 signal transmitted as [zeros(1,Lv) v1 zeros(1,Lv)], i.e., length-Lv vector centered between two length-Lv zero vectors

% User 2 signal transmitted as [zeros(1,tau) v2 zeros(1,2*Lv-tau)], i.e., length-Lv vector at arbitrary delay tau

% Generate MSK via OFFSET QPSK, filtered with a sinusoidal pulse.

\[ u = \sin(\pi/Ts \cdot (0:1/fsT:T_s-1/fsT) + \text{taus} \cdot \pi/Ts) \] % Create sinusoidal pulse shape

Nover=32;

\[ uV = \sin(\pi/2 \cdot (0:1/(Nover/2):2-1/(Nover/2))) \] % Create "template" pulse, Nover samples/Ts, to which estimate can be compared

b1=BIN01(N,0.5);    b2=BIN01(N,0.5); % Generate random binary vectors

bover1=OverN(b1/fsT); bover2=OverN(b2/fsT); % Oversample

I1=zeros(1,N*fsT+fsT);  Q1=zeros(1,N*fsT+fsT);  % Initialize I1 and Q1 components

I2=zeros(1,N*fsT+fsT);  Q2=zeros(1,N*fsT+fsT);  % Initialize I2 and Q2 components

for ii=1:2:N;       % Loop to create MSK baseband vectors
    I1((ii-1)*fsT+1:(ii+1)*fsT) = (-2*b1(ii)+1)*u;   % '0' maps to 1; '1' maps to -1
    Q1(ii*fsT+1:(ii+2)*fsT) = (-2*b1(ii+1)+1)*u;
    I2((ii-1)*fsT+1:(ii+1)*fsT) = (-2*b2(ii)+1)*u;
    Q2(ii*fsT+1:(ii+2)*fsT) = (-2*b2(ii+1)+1)*u;
end

% Generate Doppler envelope for CC interfering signal (signal 2)

Ad2c=cos(2*pi*fD2*(0:N*fsT+fsT-1)/fsT+phi2);  Ad2s=sin(2*pi*fD2*(0:N*fsT+fsT-1)/fsT+phi2);

% Generate baseband MSK signals, with Doppler envelope (only on weak signal v2)
% If NO Doppler, vk=Ak*(Ik-j*Qk), where k=1 or 2

v2=A2*(I2(1:N*fsT+fsT).*Ad2c-Q2(1:N*fsT+fsT).*Ad2s);

% For burst Asynchronous, set bA=1

bA=0;

if bA == 1
    v1=[zeros(1,Lv) v1 zeros(1,Lv)]; % Lengthen signal 1 to 3*Lv for async
    tau=floor(rand(1,1)*(2*Lv+1)); % User 2 random delay, between 0 and 2*Lv samples
    v2=[zeros(1,tau) v2 zeros(1,2*Lv-tau)]; % User 2 signal, async
    n1=(randn(1,LvA)+j*randn(1,LvA))*1; % Baseband AWGN
    rA1=v1+v2+n1; % Baseband received signal
    rs1=rA1(Lv+1:2*Lv); % Extract signal 1 subsequence from r
end
else
    v2 = cshift(v2, Randinteg(1,1,fsT-1));
    n1 = randn(1,Lv) + j*randn(1,Lv);
    rA1 = v1 + v2 + n1;
    rs1 = rA1; % Extract signal 1 subsequence from r
end

rAe = sqrt(sum(real(rs1).^2 + imag(rs1).^2)/length(rs1)); % sqrt(rI^2 + rQ^2 ) amp estimate
NMSErs = NMSErs + (1 - rAe/A1)^2/Ntrials;

% Now, amplitude estimation of user 1 signal................
I1hat = real(rs1(1:N*fsT)); Q1hat = -imag(rs1(fsT+1:N*fsT+fsT));

% Organize samples by position within each symbol--get 2*fsT samples/symbol, so 2*fsT sub-vectors
Imat = reshape(I1hat, 2*fsT, N/2); Qmat = reshape(Q1hat, 2*fsT, N/2);

Im = Imat'; Qm = Qmat'; % Each column of Im (or Qm) has samples in same position
% uhatI and uhatQ are each vectors of 2*fsT samples, estimate of pulse sin(pi*(t+taus)/(2*T)),
\nx = \frac{2*sum(abs(Im))}{N}; \quad uhatQ = \frac{2*sum(abs(Qm))}{N};
\ny = \frac{2*sum(abs(Im))}{N}; \quad uhatQ = \frac{2*sum(abs(Qm))}{N};

uest = (uhatI + uhatQ)/2; % Averaging of BOTH I and Q samples together
uestS = overN(uest/A1, Nover/Ns); % Oversample estimate for determining delay estimate tauhat

% Next need to "extract" amplitude A1 from these averaged pulse samples;
uest = (uest + uhatQ)/2; % Averaging of BOTH I and Q samples together
uestS = overN(uest/A1, Nover/Ns); % Oversample estimate for determining delay estimate tauhat

% u2 = sin(pi/Ts*(0:1/fsT:Ts-1/fsT)+tauhat*pi/Ts);
A1hat = max(uest)/max(u2);
NMSE = NMSE + (1 - A1hat/A1)^2/Ntrials;
end

p2(k) = NMSE
k = k+1;
%%This part assumes Eb/N01=6 dB and NFR=6 dB phase synchronous

j=sqrt(-1);
BER1=0; % If user-1 BER computation desired, set BER1=1; **NOT QUITE ACCURATE IN
SECTION AT BOTTOM***
fsT=4;k=1; % Normalized sampling rate, per bit (fs=sampling
frequency,T=bit time=1 here)
T=1;Ts=2*T; % Set bit and symbol durations
Ns=2*fsT; % Ns=#samples/symbol=#samples/Ts=2*fsT
Ntrials=500; % # trials over which to average amp estimate quality, for a given Eb1/N0, N, and NFS
Eb1=6; % Energy per bit to spectral noise density ratio (Eb/N0) for user 1, in dB
Eb1n=10^(Eb1/10); % Eb/N0 for user 1, numeric; (user 1 is "strong" signal)
A1=sqrt(2*Eb1n/fsT); % Amplitude of user 1 MSK signal (T=1)
NFR=6; % Near-far ratio, in dB (Eb/N0)strong/(Eb/N0)weak
A2=A1*(10^(-NFR/20)); % Amplitude of user 2 MSK signal
Eb2n=(A2^2)*fsT/2; % Eb/N0 for user 2, numeric; (user 2 is "weak" signal)
Eb2=10*log10(Eb2n); % Eb/N0 for user 2, in dB
for N=10:5:100 % Number of BITS used, must be EVEN (N/2 symbols on both I and Q)
    if mod(N,2) ~= 0; N=N+1; end

end %%%

if mod(N,2) ~= 0; N=N+1; end

for ik=1:Ntrials
    fD2=0.000001; % Doppler for co-channel signal (signal 2); set VERY low
    phi2=0; % Random phase for co-channel signal
    taus=rand(1,1)*1/fsT; % Sampling phase offset for strong signal 1, uniform between 0 and
    Ts/Ns=Ts/2/fsT=2*T/2/fsT=1/fsT
    taus=0; % Set sampling phase offset to zero for testing
    % For burst Asynchronous reception, lengthen vectors; otherwise, just circularly shift signal 2
    LvA=3*Lv; % Use the following for burst Async
% User 1 signal transmitted as \([\text{zeros}(1, \text{Lv}) \; v1 \; \text{zeros}(1, \text{Lv})]\), i.e., length-\text{Lv} vector centered between two length-\text{Lv} zero vectors

% User 2 signal transmitted as \([\text{zeros}(1, \tau) \; v2 \; \text{zeros}(1, 2\cdot\text{Lv}-\tau)]\), i.e., length-\text{Lv} vector at arbitrary delay \(\tau\)

% Generate MSK via OFFSET QPSK, filtered with a sinusoidal pulse.
\[ u = \sin(\pi/f_{Ts} \cdot (0:1/f_{Ts}\cdot Ts-1/f_{Ts})+\tau \cdot \pi/f_{Ts}) \]  
\[ \text{Nover} = 32; \]
\[ uV = \sin(\pi/2 \cdot (0:1/(\text{Nover}/2):2-1/(\text{Nover}/2))) \]  
% Create "template" pulse, Nover samples/Ts, to which estimate can be compared

b1 = \text{BIN01}(N, 0.5);  \quad b2 = \text{BIN01}(N, 0.5);  \% Generate random binary vectors
\text{bover1} = \text{OverN}(b1, f_{Ts});  \quad \text{bover2} = \text{OverN}(b2, f_{Ts});  \% Oversample
I1 = \text{zeros}(1, N\cdot f_{Ts}+f_{Ts});  \quad Q1 = \text{zeros}(1, N\cdot f_{Ts}+f_{Ts});  \% Initialize I1 and Q1 components
I2 = \text{zeros}(1, N\cdot f_{Ts}+f_{Ts});  \quad Q2 = \text{zeros}(1, N\cdot f_{Ts}+f_{Ts});  \% Initialize I2 and Q2 components

\% Loop to create MSK baseband vectors
\begin{align*}
I1((ii-1)\cdot f_{Ts}+1:(ii+1)\cdot f_{Ts}) &= (-2 \cdot b1(ii)+1) \cdot u; \quad \% \text{'0' maps to 1; \text{'}1' maps to -1}
Q1((ii+1)\cdot f_{Ts}+1:(ii+2)\cdot f_{Ts}) &= (-2 \cdot b1(ii)+1) \cdot u; \\
I2((ii-1)\cdot f_{Ts}+1:(ii+1)\cdot f_{Ts}) &= (-2 \cdot b2(ii)+1) \cdot u; \\
Q2((ii+1)\cdot f_{Ts}+1:(ii+2)\cdot f_{Ts}) &= (-2 \cdot b2(ii)+1) \cdot u;
\end{align*}

\% Generate Doppler envelope for CC interfering signal (signal 2)
\[ \text{Ad2c} = \cos(2 \cdot \pi \cdot f_{D2} \cdot [0:N\cdot f_{Ts}+f_{Ts}-1]/f_{Ts}+\phi_2); \]
\[ \text{Ad2s} = \sin(2 \cdot \pi \cdot f_{D2} \cdot [0:N\cdot f_{Ts}+f_{Ts}-1]/f_{Ts}+\phi_2); \]
% Generate baseband MSK signals, with Doppler envelope (only on weak signal v2)
% If NO Doppler, \(vk = A_k \cdot (I_k-j\cdot Q_k)\), where \(k=1\) or 2
\begin{align*}
v2 &= A2 \cdot (I2(1:N\cdot f_{Ts}+f_{Ts})).\text{Ad2c}-Q2(1:N\cdot f_{Ts}+f_{Ts}).\text{Ad2s}; \\
v2 &= v2+j \cdot A2 \cdot (-I2(1:N\cdot f_{Ts}+f_{Ts})).\text{Ad2s}+Q2(1:N\cdot f_{Ts}+f_{Ts}).\text{Ad2c}; \\
v1 &= A1 \cdot (I1(1:N\cdot f_{Ts}+f_{Ts})).j \cdot Q1(1:N\cdot f_{Ts}+f_{Ts});
\end{align*}
% For burst Asynchronous, set \(bA = 1\)
\(bA = 0;\)
if \(bA = 1\)
\begin{align*}
v1 &= \text{zeros}(1, \text{Lv}) \; v1 \; \text{zeros}(1, \text{Lv});  \% \text{Lengthen signal 1 to 3} \cdot \text{Lv} \text{ for async} \\
\tau &= \text{floor}(\text{rand}(1, 1) \cdot (2 \cdot \text{Lv}+1));  \% \text{User 2 random delay, between 0 and 2} \cdot \text{Lv} \text{ samples} \\
v2 &= \text{zeros}(1, \text{Lv}) \; v2 \; \text{zeros}(1, 2\cdot\text{Lv}-\tau);  \% \text{User 2 signal, async} \\
n1 &= (\text{randn}(1, \text{LvA})+j \cdot \text{randn}(1, \text{LvA}))*1;  \% \text{Baseband AWGN} \\
rA1 &= v1+v2+n1;  \% \text{Baseband received signal} \\
rS1 &= rA1(\text{Lv}+1:2\cdot\text{Lv});  \% \text{Extract signal 1 subsequence from r}
else
    v2=cshift(v2, Randinteg(1,1,fsT-1));
    n1=randn(1,Lv)+j*randn(1,Lv);
    rA1=v1+v2+n1;
    rs1=rA1;                                    % Extract signal 1 subsequence from r
end
rAe=sqrt(sum(real(rs1).^2+imag(rs1).^2)/length(rs1));   % sqrt(rI^2 + rQ^2 ) amp estimate
NMSErs=NMSErs+(1-rAe/A1)^2/Ntrials;
% Now, amplitude estimation of user 1 signal.................
I1hat=real(rs1(1:N*fsT)); Q1hat=-imag(rs1(fsT+1:N*fsT+fsT));
% Organize samples by position within each symbol--get 2*fsT samples/symbol, so 2*fsT sub-vectors
Imat=reshape(I1hat,2*fsT,N/2); Qmat=reshape(Q1hat,2*fsT,N/2);
Im=Imat'; Qm=Qmat';         % Each column of Im (or Qm) has samples in same position
% uhatI and uhatQ are each vectors of 2*fsT samples, estimate of pulse sin(pi*(t+taus)/(2*T)),
t=k*fsT, k=0,1,...2*fsT-1
    uhatI=2*sum(abs(Im))/N; uhatQ=2*sum(abs(Qm))/N;
    % Next need to "extract" amplitude A1 from these averaged pulse samples;
    uest=(uhatI+uhatQ)/2;   % Averaging of BOTH I and Q samples together
    uestS=overN(uest/A1,Nover/Ns); % Oversample estimate for determining delay estimate tauhat
    (assumes "perfect" scaling with true A1)
    uev=abs(uestS-uV);     % Compare oversampled estimate with "template" pulse uV
    [du,iu]=min(uev);   % Find index of minimum difference between estimate & template
    mdelay=mod(iu,Nover/Ns); % NOTE: NOT CERTAIN OF ACCURACY of the "if" function here
    if mdelay == 0;
        mdelay;
        tauh=((Nover/Ns)-1)/Nover;
    else
        tauh=(mod(iu,Nover/Ns)-1)/Nover;% tauh is quantized version of tauhat, in increments of 1/Nover
    end
    tauhat=Ts*tauh;            % Final estimate of taus
    u2=sin(pi/Ts*(0:1/fsT:Ts-1/fsT)+tauh*pi/Ts);
    A1hat=max(uest)/max(u2);
    NMSE=NMSE+(1-A1hat/A1)^2/Ntrials;
end
p3(k)=NMSE
k=k+1;
This part assumes Eb/N01=10 dB and NFR=3 dB phase synchronuous

\[
j = \sqrt{-1};
\]

\[
\text{BER1}=0; \quad \text{If user-1 BER computation desired, set BER1}=1; \quad \text{**NOT QUITE ACCURATE IN SECTION AT BOTTOM***}
\]

\[
f_T=4; k=1; \quad \text{fsampling rate, per bit (fs= sampling frequency, T= bit time = 1 here)}
\]

\[
T=1; T_s=2*T; \quad \text{Set bit and symbol durations}
\]

\[
N_s=2*f_T; \quad \text{Ns=#samples/symbol=#samples/Ts=2*fTs}
\]

\[
N_trials=500; \quad \text{# trials over which to average amp estimate quality, for a given Eb1}/N0, N, and NFS
\]

\[
Eb1=10; \quad \text{Energy per bit to spectral noise density ratio (Eb/N0) for user 1, in dB}
\]

\[
Eb1n=10^{(Eb1/10)}; \quad \text{Eb/N0 for user 1, numeric; (user 1 is "strong" signal)}
\]

\[
A1=\sqrt{2*Eb1n/fTs}; \quad \text{Amplitude of user 1 MSK signal (T=1)}
\]

\[
NFR=3; \quad \text{Near-far ratio, in dB (Eb/N0)strong/(Eb/N0)weak}
\]

\[
A2=A1*(10^{(-NFR/20)}); \quad \text{Amplitude of user 2 MSK signal}
\]

\[
Eb2n=(A2^2)*fTs/2; \quad \text{Eb/N0 for user 2, numeric; (user 2 is "weak" signal)}
\]

\[
Eb2=10*log10(Eb2n); \quad \text{Eb/N0 for user 2, in dB}
\]

for \(N=10:5:100\); \quad \text{Number of BITS used, must be EVEN (N/2 symbols on both I and Q)}

\[
\text{if mod(N,2) }\approx 0; \quad N=N+1; \quad \text{end}
\]

\[
t=0:1/fTs:N-1/fTs; \quad \text{Create time vector for plots}
\]

\[
L_v=fTs*N; \quad \text{Length of complex transmitted vectors v1 and v2}
\]

\[
L_v=L_v+fTs; \quad \text{Extend vector by one bit, fsT samples}
\]

\[
\text{NMSE}=0; \quad \text{Initialize the normalized mean-square amplitude estimation error}
\]

\[
\text{NMSErs}=0; \quad \text{Initialize normalized MS amp est error for simple sqrt(rI^2+rQ^2) estimator}
\]

\[
\text{for ik=1:Ntrials}
\]

\[
fD2=0.000001; \quad \text{Doppler for co-channel signal (signal 2); set VERY low}
\]

\[
\phi2=0; \quad \text{Random phase for co-channel signal}
\]

\[
\text{taus} = \text{rand}(1,1)*1/fTs; \quad \text{Sampling phase offset for strong signal 1, uniform between 0 and}
\]

\[
Ts/Ns=T_s/2/fTs=2*T_s/2/fTs=1/fTs
\]

\[
\text{taus}=0; \quad \text{Set sampling phase offset to zero for testing}
\]

\[
\text{if For burst Asynchronous reception, lengthen vectors; otherwise, just circularly shift signal 2}
\]

\[
L_vA=3*L_v; \quad \text{Use the following for burst Async}
\]
% User 1 signal transmitted as [zeros(1,Lv) v1 zeros(1,Lv)], i.e., length-Lv vector centered between two length-Lv zero vectors
% User 2 signal transmitted as [zeros(1,tau) v2 zeros(1,2*Lv-tau)], i.e., length-Lv vector at arbitrary delay tau

% Generate MSK via OFFSET QPSK, filtered with a sinusoidal pulse.
u=sin(pi/Ts*(0:1/fsT:Tsv-1/fsT)+taus*pi/Ts);  % Create sinusoidal pulse shape
Nover=32;
uV=sin(pi/2*(0:1/(Nover/2):2-1/(Nover/2)));      % Create "template" pulse, Nover samples/Ts, to which estimate can be compared
b1=BIN01(N,0.5);        b2=BIN01(N,0.5);        % Generate random binary vectors
bover1=OverN(b1,fsT);   bover2=OverN(b2,fsT);   % Oversample
I1=zeros(1,N*fsT+fsT);  Q1=zeros(1,N*fsT+fsT);  % Initialize I1 and Q1 components
I2=zeros(1,N*fsT+fsT);  Q2=zeros(1,N*fsT+fsT);  % Initialize I2 and Q2 components
for ii=1:2:N;       % Loop to create MSK baseband vectors
    I1((ii-1)*fsT+1:(ii+1)*fsT) = (-2*b1(ii)+1)*u;   % '0' maps to 1; '1' maps to -1
    Q1(ii*fsT+1:(ii+2)*fsT) = (-2*b1(ii+1)+1)*u;
    I2((ii-1)*fsT+1:(ii+1)*fsT) = (-2*b2(ii)+1)*u;
    Q2(ii*fsT+1:(ii+2)*fsT) = (-2*b2(ii+1)+1)*u;
end

% Generate Doppler envelope for CC interfering signal (signal 2)
Ad2c=cos(2*pi*fD2*[0:N*fsT+fsT-1]/fsT+phi2);
Ad2s=sin(2*pi*fD2*[0:N*fsT+fsT-1]/fsT+phi2);
% Generate baseband MSK signals, with Doppler envelope (only on weak signal v2)
% If NO Doppler, vk=Ak*(Ik-j*Qk), where k=1 or 2
v2=-Ad2s;v1=A1*(I1(1:N*fsT+fsT)-j*Q1(1:N*fsT+fsT));

% For burst Asynchronous, set bA=1
bA=0;
if bA == 1
    v1=[zeros(1,Lv) v1 zeros(1,Lv)];       % Lengthen signal 1 to 3*Lv for async
    tau=floor(rand(1,1)*(2*Lv+1));        % User 2 random delay, between 0 and 2*Lv samples
    v2=[zeros(1,tau) v2 zeros(1,2*Lv-tau)]; % User 2 signal, async
    n1=(randn(1,LvA)+j*randn(1,LvA))*1;    % Baseband AWGN
    rA1=v1+v2+n1;                          % Baseband received signal
    rs1=rA1(Lv+1:2*Lv);                    % Extract signal 1 subsequence from r
else
    v2=shift(v2,randinteg(1,1,fsT-1));
    n1=randn(1,Lv)+j*randn(1,Lv);
    rA1=v1+v2+n1;
    rs1=rA1;               % Extract signal 1 subsequence from r
end
rAc=sqrt(mean(real(rs1).^2+imag(rs1).^2)/length(rs1)); % sqrt(rI^2 + rQ^2) amp estimate
NMSErs=NMSErs+(1-rAc/A1)^2/Ntrials;

% Now, amplitude estimation of user 1 signal..............
I1hat=real(rs1(1:N*fsT)); Q1hat=-imag(rs1(fsT+1:N*fsT+fsT));
% Organize samples by position within each symbol—get 2*fsT samples/symbol, so 2*fsT sub-vectors
Imat=reshape(I1hat,2*fsT,N/2); Qmat=reshape(Q1hat,2*fsT,N/2);
Im=Imat'; Qm=Qmat';       % Each column of Im (or Qm) has samples in same position
% uhatI and uhatQ are each vectors of 2*fsT samples, estimate of pulse sin(pi*(t+tau)/T)
   t=k*fsT, k=0,1,...2*fsT-1
   uhatI=2*sum(abs(Im))/N; uhatQ=2*sum(abs(Qm))/N;
% Next need to "extract" amplitude A1 from these averaged pulse samples
uest=(uhatI+uhatQ)/2;     % Averaging of BOTH I and Q samples together
uestS=overN(uest/A1,Nover/Ns); % Oversample estimate for determining delay estimate tauhat
(assumes "perfect" scaling with true A1)
uev=abs(uestS-uV);        % Compare oversampled estimate with "template" pulse uV
[du,iu]=min(uev);          % Find index of minimum difference between estimate & template
mdelay=mod(iu,Nover/Ns);  % NOTE: NOT CERTAIN OF ACCURACY of the "if" function here
if mdelay == 0;
    mdelay;
    tauh= ((Nover/Ns)-1)/Nover;
else
    tauh= (mod(iu,Nover/Ns)-1)/Nover;% tauh is quantized version of tauhat, in increments of 1/Nover
end
tauhat=Ts*tauh;           % Final estimate of taus
u2=sin(pi/Ts*(0:1/fsT:Ts-1/fsT)+tauhat*pi/Ts);
A1hat=max(uest)/max(u2);
NMSE=NMSE+(1-A1hat/A1)^2/Ntrials;
end
p4(k)=NMSE
k=k+1;
%This part assumes Eb/N01=10 dB and NFR=6 dB- phase synchronous

\[ j = \sqrt{-1}; \]

BER1=0; \hspace{1cm} \% If user-1 BER computation desired, set BER1=1; \hspace{1cm} **NOT QUITE ACCURATE IN SECTION AT BOTTOM**

fsT=4;k=1; \hspace{1cm} \% Normalized sampling rate, per bit (fs=sampling frequency, T=bit time=1 here)

T=1;Ts=2*T; \hspace{1cm} \% Set bit and symbol durations

Ns=2*fsT; \hspace{1cm} \% Ns=#samples/symbol=#samples/Ts=2*fsT

Ntrials=500; \hspace{1cm} \% # trials over which to average amp estimate quality, for a given Eb1/N0, N, and NFS

Eb1=10; \hspace{1cm} \% Energy per bit to spectral noise density ratio (Eb/N0) for user 1, in dB

Eb1n=10^(Eb1/10); \hspace{1cm} \% Eb/N0 for user 1, numeric; (user 1 is "strong" signal)

A1=sqrt(2*Eb1n/fsT); \hspace{1cm} \% Amplitude of user 1 MSK signal (T=1)

NFR=6; \hspace{1cm} \% Near-far ratio, in dB (Eb/N0)strong/(Eb/N0)weak

A2=A1*(10^(-NFR/20)); \hspace{1cm} \% Amplitude of user 2 MSK signal

Eb2n=(A2^2)*fsT/2; \hspace{1cm} \% Eb/N0 for user 2, numeric; (user 2 is "weak" signal)

Eb2=10*log10(Eb2n); \hspace{1cm} \% Eb/N0 for user 2, in dB

for N=10:5:100 \hspace{1cm} % Number of BITS used, must be EVEN (N/2 symbols on both I and Q)

    if mod(N,2) ~= 0; N=N+1; end

    t=0:1/fsT:N-1/fsT; \hspace{1cm} \% Create time vector for plots

    Lv=fsT*N; \hspace{1cm} \% Length of complex transmitted vectors v1 and v2

    Lv=Lv+fsT; \hspace{1cm} \% Extend vector by one bit, fsT samples

    NMSE=0; \hspace{1cm} \% Initialize the normalized mean-square amplitude estimation error

    NMSErs=0; \hspace{1cm} \% Initialize normalized MS amp est error for simple sqrt(rI^2+rQ^2) estimator

    for ik=1:Ntrials

        fD2=0.000001; \hspace{1cm} \% Doppler for co-channel signal (signal 2); set VERY low

        phi2=0; \hspace{1cm} \% Random phase for co-channel signal

        taus=rand(1,1)*1/fsT; \hspace{1cm} \% Sampling phase offset for strong signal 1, uniform between 0 and

        Ts/Ns=Ts/2/fsT=2*T/2/fsT=1/fsT

        taus=0; \hspace{1cm} \% Set sampling phase offset to zero for testing

        % For burst Asynchronous reception, lengthen vectors; otherwise, just circularly shift signal 2
LvA = 3 * Lv;  % Use the following for burst Async

% User 1 signal transmitted as [zeros(1, Lv) v1 zeros(1, Lv)], i.e., length-Lv vector centered between
two length-Lv zero vectors

% User 2 signal transmitted as [zeros(1, tau) v2 zeros(1, 2 * Lv - tau)], i.e., length-Lv vector at arbitrary
delay tau

% Generate MSK via OFFSET QPSK, filtered with a sinusoidal pulse.
u = sin(pi/Ts*(0:1/fsT:T0-1/fsT)+taus*pi/Ts);  % Create sinusoidal pulse shape
Nover = 32;
uV = sin(pi/2*(0:1/(Nover/2):2-1/(Nover/2)));  % Create "template" pulse, Nover samples/Ts, to
which estimate can be compared

b1 = BIN01(N, 0.5); b2 = BIN01(N, 0.5);  % Generate random binary vectors
bover1 = OverN(b1, fsT); bover2 = OverN(b2, fsT);  % Oversample
I1 = zeros(1, N*fsT + fsT); Q1 = zeros(1, N*fsT + fsT);  % Initialize I1 and Q1 components
I2 = zeros(1, N*fsT + fsT); Q2 = zeros(1, N*fsT + fsT);  % Initialize I2 and Q2 components

for ii = 1:2:N
% Loop to create MSK baseband vectors
I1((ii-1)*fsT+1:(ii+1)*fsT) = (-2*b1(ii)+1)*u;  % '0' maps to 1; '1' maps to -1
Q1(ii*fsT+1:(ii+2)*fsT) = (-2*b1(ii+1)+1)*u;
I2((ii-1)*fsT+1:(ii+1)*fsT) = (-2*b2(ii)+1)*u;
Q2(ii*fsT+1:(ii+2)*fsT) = (-2*b2(ii+1)+1)*u;
end

% Generate Doppler envelope for CC interfering signal (signal 2)
Ad2c = cos(2*pi*fD2*[0:N*fsT+fsT-1]/fsT+phi2);
Ad2s = sin(2*pi*fD2*[0:N*fsT+fsT-1]/fsT+phi2);

% Generate baseband MSK signals, with Doppler envelope (only on weak signal v2)
% If NO Doppler, vk = Ak*(Ik-j*Qk), where k = 1 or 2
v2 = A2*(I2(1:N*fsT+fsT).*Ad2c-Q2(1:N*fsT+fsT).*Ad2s);
v2 = v2+j*A2*(-I2(1:N*fsT+fsT).*Ad2s+Q2(1:N*fsT+fsT).*Ad2c);
v1 = A1*(I1(1:N*fsT+fsT)+j*Q1(1:N*fsT+fsT));

% For burst Asynchronous, set bA = 1
bA = 0;

if bA == 1
v1 = [zeros(1, Lv) v1 zeros(1, Lv)];  % Lengthen signal 1 to 3*Lv for async
tau = floor(rand(1, 1)*(2*Lv+1));  % User 2 random delay, between 0 and 2*Lv samples
v2 = [zeros(1, tau) v2 zeros(1, 2*Lv - tau)];  % User 2 signal, async
n1 = (randn(1, LvA)+j*randn(1, LvA))*1;  % Baseband AWGN
rA1 = v1+v2+n1;  % Baseband received signal
rs1=rA1(Lv+1:2*Lv); % Extract signal 1 subsequence from r
else
  v2=cshift(v2,randinteg(1,1,fsT-1));
  n1=randn(1,Lv)+j*randn(1,Lv);
  rA1=v1+v2+n1;
  rs1=rA1; % Extract signal 1 subsequence from r
end
rAe=sqrt(sum(real(rs1).^2+imag(rs1).^2)/length(rs1)); % sqrt(rI^2 + rQ^2 ) amp estimate
NMSERs=NMSERs+(1-rAe/A1)^2/Ntrials;
% Now, amplitude estimation of user 1 signal..............
I1hat=real(rs1(1:N*fsT)); Q1hat=-imag(rs1(fsT+1:N*fsT+fsT));
% Organize samples by position within each symbol--get 2*fsT samples/symbol, so 2*fsT sub-vectors
Imat=reshape(I1hat,2*fsT,N/2); Qmat=reshape(Q1hat,2*fsT,N/2);
Im=Imat'; Qm=Qmat'; % Each column of Im (or Qm) has samples in same position
% uhatI and uhatQ are each vectors of 2*fsT samples, estimate of pulse sin(pi*(t+taus)/(2*T)),
t=k*fsT, k=0,1,...2*fsT-1
uhatI=2*sum(abs(Im))/N; uhatQ=2*sum(abs(Qm))/N;
% Next need to "extract" amplitude A1 from these averaged pulse samples;
uest=(uhatI+uhatQ)/2; % Averaging of BOTH I and Q samples together
uestS=overN(uest/A1,Nover/Ns); % Oversample estimate for determining delay estimate tauhat
(assumes "perfect" scaling with true A1)
uev=abs(uestS-uV); % Compare oversampled estimate with"template" pulse uV
[du,iu]=min(uev); % Find index of minimum difference between estimate & template
mdelay=mod(iu,Nover/Ns); % NOTE: NOT CERTAIN OF ACCURACY of the "if" function here
if mdelay == 0;
  mdelay;
  tauh=((Nover/Ns)-1)/Nover;
else
  tauh=(mod(iu,Nover/Ns)-1)/Nover;% tauh is quantized version of tauhat, in increments of 1/Nover
end
tauhat=Ts*tauh; % Final estimate of tau
u2=sin(pi/Ts*(0:1/fsT:Tsl-1/fsT)+tauhat*pi/Ts);
A1hat=max(uest)/max(u2);
NMSE=NMSE+(1-A1hat/A1)^2/Ntrials;
end
p5(k)=NMSE
k=k+1;
end
q(1)=10;q(2)=15;q(4)=25;q(6)=35;q(8)=45;q(10)=55;q(12)=65;q(14)=75;q(16)=85;q(18)=95;q(3)=20;q(5)=30;q(7)=40;q(9)=50;q(11)=60;q(13)=70;q(15)=80;q(17)=90;q(19)=100;
plot(q,p,'-or',q,p1,'-xk',q,p2,q,p3,q,p4,'-.or',q,p5,'-.xk',q,p6,'
grid on;
xlabel('N-->');
ylabel('NMSE-->');
title('NMSE vs N for E_b_1/N_o=6 dB -NFR=3 dB and E_b_1/N_o=10 dB -NFR=6 dB ');
legend('phase asynchronous','phase synchronous');

%%% The above program was also run by replacing the sinusoidal waveform with 2 rectangular pulse streams

% Amplitude Estimation of MSK in Co-Channel Interference
% The following program compares the absolute value estimate, root mean square estimate,
% estimate using formula (neglecting approximations) for a rectangular pulse stream in AWGN.
% The NMSE is plotted against the %E_b/N_0 for a particular value of N.
% Ravikanth Ekanthalingam
% Ohio University,Athens.
%================================================================================================

clear all;
close all;
cle;

sn0=0;sn1=0;sn2=0;sn1fin=0;sn2fin=0;sn02=0;sn3=0;sn1n=0;sn02n=0;sn3n=0;error1=0;error2=0;error3=0;
for k=1:1:500;
    r=rand(1,100);
end

for k=1:1:500;
    n=rndn(1,50);
    n0=2*var(n);
    N=input('Enter the value of N: ');

for k=1:1:500;
    r=rndn(1,100);
end
for EbNodb=1:1:10;
    EbNo=10^(EbNodb/10);
    A=sqrt(EbNo*n0)
    for j=1:1:N
        if (r(j)>0.5)
            p(j)=A;
        else
            p(j)=-A;
        end
        sn(j)=p(j)+n(j);
    end

    for i=1:1:EbNodb
        sn1(i+1)=sn1(i)+4*(sn(i)*sn(i)); % estimation using root mean square of amplitude
        sn02(i+1)=sn02(i)+abs(sn(i)); % estimation using absolute value of amplitude
        sn3(i+1)=sn3(i)+abs(sn(i))*(tanh((sn(i)*p(i))/(n0/2))); % estimation using the formula
        i=i+1;
    end

    sn1n(EbNodb)=sqrt(sn1(EbNodb+1)/EbNodb);
    sn02n(EbNodb)=sn02(EbNodb+1)/EbNodb;
    sn3n(EbNodb)=sn3(EbNodb+1)/EbNodb;
    error1(k,EbNodb)=(1-(sn1n(EbNodb)/abs(p(EbNodb))))^2;
    error2(k,EbNodb)=(1-(sn02n(EbNodb)/abs(p(EbNodb))))^2;
    error3(k,EbNodb)=(1-(sn3n(EbNodb)/abs(p(EbNodb))))^2;
end

sn1n1=mean(error1);
sn02n1=mean(error2);
sn03n1=mean(error3);
p=sn1n1(2);
EbNo=1:1:10;
plot(EbNo,sn1n1)
hold on;
plot(EbNo,sn03n1,'*-k')
hold on;
plot(EbNo,sn02n1,'-.r');
hold on;
grid on;
xlabel(' EbNo ');
ylabel(' NMSE');
legend('rms estimate','absolute value estimate','estimate using formula');
axis([2 10 0 p ])
hold on;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% The following program is a practical application of a system consisting of 2 users in the presence of
% AWGN. The bit error probability of the weak user is plotted against the Near Far Ratio (NFR) to test
the efficiency of the estimator .

clear all;
close all;
cle;
j=sqrt(-1);
fsT=4; % Normalized sampling rate, per bit (fs=sampling frequency,T=bit
time=1 here)
T=1;Ts=2*T; % Set bit and symbol durations
Ns=2*fsT; % Ns=#samples/symbol=#samples/Ts=2*fsT
Ntrials=1;
Pbstar=zeros(1,30);
for i=1:1:300
  j=sqrt(-1);
  fsT=4; % Normalized sampling rate, per bit (fs=sampling frequency,
  % T=bit time =1 here)
  Eb2=2; % Energy per bit to spectral noise density ratio% (Eb/N0) for user 2, in dB
  Eb2n=10*(Eb2/10); % Eb/N0 for user 2, numeric
  A2=sqrt(2*Eb2n/fsT);% Amplitude of user 2 MSK signal (Tb=1)
  for NFR=1:1:30; % Near-far ratio, in dB (Eb/N0)strong/(Eb/N0)weak
    A1=A2*(10^(NFR/20));% Amplitude of user 1 MSK signal
    Eb1n=(A1^2)*fsT/2; % Eb/N0 for user 1, numeric
    Eb1=10*log10(Eb1n); % Eb/N0 for user 1, in dB
    Na=1; % Na = number of antennas used; must be either 1 or 2
\[ E_{b1\text{eff}(NFR)} = 10 \log_{10} \left( \frac{E_{b1n}}{(1+E_{b2n})} \right) \]; % Effective Eb/N0 for user 1, Eb1/(N0+Eb2)  \
\[ E_{b2\text{eff}(NFR)} = 10 \log_{10} \left( \frac{E_{b2n}}{(1+E_{b1n})} \right) \]; % Effective Eb/N0 for user 2, Eb2/(N0+Eb1)  

fD1=0.01; % Doppler frequency of strong (user 1) signal, relative to Tb, i.e., fD=fDoppler*Tb  
phi1=2*pi*rand(1,1); % Carrier phase of strong signal  
taus=rand(1,1)*1/fsT; % Sampling phase offset for strong signal 1, uniform between 0 and 1 \( T_s/N_s = T_s/2/f_sT = 2T/2/f_sT = 1/f_sT \)  

Ae=0; % Percentage amplitude estimation error on strong signal, used to assess effect of imperfect amp estimation on IC  
f1e=0; % Percentage frequency estimation error on strong signal  
N=400; % Number of bits used, must be EVEN. Generally, 200 < N < 800.  

if mod(N,2) ~= 0; N=N+1; end  

Lv=f_sT*N; % Length of complex transmitted vectors v1 and v2  
Lv=Lv+f_sT; % Extend vector by one bit, f_sT samples  
LvA=3*Lv; % Allow for asynchronous reception, as follows:  

t=0:1/f_sT:N-1/f_sT; % Create time vector for plots  
% Generate MSK via OFFSET QPSK, filtered with a sinusoidal pulse.  

\[ u = \sin(\pi/T_s*(0:1/f_sT:T_s-1/f_sT)+\tau_{us}/T_s) \]; % Create sinusoidal pulse shape  
Nover=32;  
uV=\sin(\pi/2*(0:1/(Nover/2):2-1/(Nover/2))); % Create "template" pulse, Nover samples/Ts, to which estimate can be compared  

b1=BIN01(N,0.5);  
b2=BIN01(N,0.5); % Generate random binary vectors  
bover1=OverN(b1,f_sT);  
bover2=OverN(b2,f_sT); % Oversample  
I1=zeros(1,N*f_sT+f_sT);  
Q1=zeros(1,N*f_sT+f_sT); % Initialize I1 and Q1 components  
I2=zeros(1,N*f_sT+f_sT);  
Q2=zeros(1,N*f_sT+f_sT); % Initialize I2 and Q2 components  

for ii=1:2:N; % Loop to create MSK baseband vectors  
    I1((ii-1)*f_sT+1:(ii+1)*f_sT) = (-2*b1(ii)+1)*u; % '0' maps to 1; '%1' maps to -1  
    Q1((ii-1)*f_sT+1:(ii+1)*f_sT) = (-2*b1(ii+1)+1)*u;  
    I2((ii-1)*f_sT+1:(ii+1)*f_sT) = (-2*b2(ii)+1)*u;  
    Q2((ii-1)*f_sT+1:(ii+1)*f_sT) = (-2*b2(ii+1)+1)*u;
end

% Generate Doppler envelope for strong signal
Ad1c=cos(2*pi*fD1*[0:N*fsT+fsT-1]/fsT+phi1);
Ad1s=sin(2*pi*fD1*[0:N*fsT+fsT-1]/fsT+phi1);

% Generate baseband MSK signals, with Doppler envelope (so far Doppler only on strong signal)
v1=A1*(I1(1:N*fsT+fsT)-Q1(1:N*fsT+fsT));
v1=v1+j*A1*(-I1(1:N*fsT+fsT)-Q1(1:N*fsT+fsT));
v2=A2*(I2(1:N*fsT+fsT)-j*Q2(1:N*fsT+fsT));

% For burst Asynchronous, set bA=1
bA=0;
if bA == 1
  v1=[zeros(1,Lv) v1 zeros(1,Lv)];       % Lengthen signal 1 to 3*Lv for async
  tau=floor(rand(1,1)*(2*Lv+1));        % User 2 random delay, between 0 and 2*Lv samples
  v2=[zeros(1,tau) v2 zeros(1,2*Lv-tau)]; % User 2 signal, async
  n1=(randn(1,LvA)+j*randn(1,LvA))*1;    % Baseband AWGN
  rA1=v1+v2+n1;                          % Baseband received signal
  rs1=rA1(Lv+1:2*Lv);                    % Extract signal 1 subsequence from r
else
  v2=cshift(v2,randinteg(1,1,fsT-1));
  n1=randn(1,Lv)+j*randn(1,Lv);
  rA1=v1+v2+n1;
  rs1=rA1;                               % Extract signal 1 subsequence from r
  tau=0;
end

rAe=sqrt(sum(real(rs1).^2+imag(rs1).^2)/length(rs1));   % sqrt(rI^2 + rQ^2 ) amp estimate

I1hat=real(rs1(1:N*fsT)); Q1hat=-imag(rs1(1:N*fsT+1:N*fsT+fsT));

% Organize samples by position within each symbol--get 2*fsT samples/symbol, so 2*fsT sub-vectors
Imat=reshape(I1hat,2*fsT,N/2); Qmat=reshape(Q1hat,2*fsT,N/2);
Im=Imat'; Qm=Qmat'; % Each column of Im (or Qm) has samples in same position

% uhatI and uhatQ are each vectors of 2*fsT samples, estimate of pulse sin(pi*(t+taus)/(2*T)),
t=k*fsT, k=0,1,...2*fsT-1
uhatI=2*sum(abs(Im))/N; uhatQ=2*sum(abs(Qm))/N;
% Next need to "extract" amplitude A1 from these averaged pulse samples;
uest=(uhatI+uhatQ)/2; % Averaging of BOTH I and Q samples together
uestS=overN(uest/A1,Nover/Ns); % Oversample estimate for determining delay estimate tauhat
(assumes "perfect" scaling with true A1)
uev=abs(uestS-uV); % Compare oversampled estimate with"template" pulse uV
[du,iu]=min(uev); % Find index of minimum difference between estimate & template
mdelay=mod(iu,Nover/Ns); % NOTE: NOT CERTAIN OF ACCURACY of the "if" function here
if mdelay == 0;
    mdelay;
    tauh=((Nover/Ns)-1)/Nover;
else
    tauh=(mod(iu,Nover/Ns)-1)/Nover; % tauh is quantized version of tauhat, in increments of 1/Nover
end
tauhat=Ts*tauh; % Final estimate of tau
u2=sin(pi/Ts*(0:1/fsT:Ts-1/fsT)+tauhat*pi/Ts);
A1hat=max(uest)/max(u2);

if BER1 == 1; % Compute user 1 Pb estimate if flag set
  Pb1=0; Pb2=0; % Initialize Pb's
  Ih11=zeros(1,N*fsT); Qh11=zeros(1,N*fsT+fsT); % Initialize I1 and Q1 component estimates
  Ih12=zeros(1,N*fsT); Qh12=zeros(1,N*fsT+fsT); % Initialize I2 and Q2 component estimates
  % Next, receiver correlation and detection for signal 1, assuming estimated carrier freq & phase, &
  perfect symbol timing
  for ir=1:2:N
      Ih11((ir-1)*fsT+1:(ir+1)*fsT)=u.*real(rs1((ir-1)*fsT+1:(ir+1)*fsT))*sqrt(2);
      Ihat11((ir+1)/2)=sign(sum(Ih11((ir-1)*fsT+1:(ir+1)*fsT))); %I-channel estimate of signal 1, hard decision
      Ihat11((ir+1)/2)=sum(Ih11((ir-1)*fsT+1:(ir+1)*fsT)); %I-channel estimate of signal 1, soft decision
      Qh11((ir+1)*fsT+1:(ir+2)*fsT)=-u.*imag(rs1((ir+1)*fsT+1:(ir+2)*fsT))*sqrt(2);
      Qhat11((ir+1)/2)=sign(sum(Qh11((ir+1)*fsT+1:(ir+2)*fsT))); %Q-channel estimate of signal 1, hard decision
      Qhat11((ir+1)/2)=sum(Qh11((ir+1)*fsT+1:(ir+2)*fsT)); % Q-channel estimate of signal 1, soft decision
  end
  bh11=LEAVE2(Ihat11,Qhat11); % Interleave soft I and Q estimates for signal 1
  bh1=(1-sign(bh11))/2; % Hard decision estimates for signal 1
bh1H = sign(bh1); % Hard bit decisions on signal 1
bhat1 = (1 - bh1H) / 2; % Convert +/-1 values to 0/1 values
er1 = abs(b1 - bhat1); % Count bit errors, and compute Pb1 estimate
Pb1 = sum(er1) / N
end

% Next, IC: remodulate estimated user 1 bits and subtract from r, then %demodulate user 2, on each antenna separately
I1r1 = zeros(1, Lv); Q1r1 = zeros(1, Lv); % Initialize reconstructed I1 and %Q1 components, antenna 1
% I1r2 = zeros(1, Lv); Q1r2 = zeros(1, Lv); % Initialize reconstructed I1 and %Q1 components, antenna 2
for ic = 1:2:N; % Loop to remodulate user 1 bit estimates
I1r1((ic - 1) * fsT + 1:(ic + 1) * fsT) = (-2 * bh1(ic) + 1) * u;
Q1r1(ic * fsT + 1:(ic + 2) * fsT) = (-2 * bh1(ic + 1) + 1) * u;
% I1r2((ic - 1) * fsT + 1:(ic + 1) * fsT) = (-2 * bhat1A2(ic) + 1) * u;
% Q1r2(ic * fsT + 1:(ic + 2) * fsT) = (-2 * bhat1A2(ic + 1) + 1) * u;
end
% A1h = A1 + ((2 * BIN01(1, 0.5) - 1) * Ae / 100) * A1; % Introduce strong-signal amplitude error of Ae
% Reconstructed user 1 baseband MSK signal, using estimated amp & gain,
% ant 1, with estimated Doppler
r1 = r1 - v1r1; % IC the user 1 signal from the first-stage %received signal, ant 1
rICs1 = r1;
for i2 = 1:2:N
lh21((i2 - 1) * fsT + 1:(i2 + 1) * fsT) = u * real(rICs1((i2 - 1) * fsT + 1:(i2 + 1) * fsT)) * sqrt(2);
Ihat21((i2 + 1) / 2) = sum(lh21((i2 - 1) * fsT + 1:(i2 + 1) * fsT)); % Antenna 1 I-channel estimate of signal 2, soft decision
Qh21((i2 + 1) / 2) = -u * imag(rICs1((i2 - 1) * fsT + 1:(i2 + 1) * fsT)) * sqrt(2);
Qhat21((i2 + 1) / 2) = sum(Qh21((i2 + 1) / 2)); % Antenna 1 Q-channel estimate of signal 2, soft decision
end
bh21 = LEAVE2(Ihat21, Qhat21); % Interleave soft I and Q estimates for signal 2, antenna 1
bhat2A1 = (1 - sign(bh21)) / 2; % Hard decision estimates for signal 2, antenna 1
bh2=bf21; % Combine soft estimates of signal 2 from both antennas, then make hard decision.

Current combining method is MRC

bh2H = sign(bh2); % Hard bit decisions on signal 2, after IC
bhat2 = (1-bh2H)/2; % Convert +/-1 values to 0/1 values
er2 = abs(b2-bhat2); % Count bit errors, and compute Pb2 estimate
Pb2(NFR) = sum(er2)/N;
NFR = NFR + 1;
end

Pbstar = Pbstar + Pb2;
end
avg = Pbstar/300;
disp('the values of Eb2 in dB are');
disp(avg);

j = sqrt(-1);
BER1 = 1; % If user-1 BER computation desired, set BER1 = 1; **NOT QUITE ACCURATE IN SECTION AT BOTTOM**

fsT = 4; % Normalized sampling rate, per bit (fs = sampling frequency, T = bit time = 1 here)
T = 1; Ts = 2*T; % Set bit and symbol durations
Ns = 2*fsT; % Ns = #samples/symbol = #samples/Ts = 2*fsT
Ntrials = 1;
Pbstar2 = zeros(1,30);
for i = 1:1:300
    j = sqrt(-1);
    fsT = 4; % Normalized sampling rate, per bit (fs = sampling frequency, T = bit time = 1 here)
    Eb2 = 2; % Energy per bit to spectral noise density ratio (Eb/N0) for user 2, in dB
    Eb2n = 10^(Eb2/10); % Eb/N0 for user 2, numeric
    A2 = sqrt(2*Eb2n/fsT); % Amplitude of user 2 MSK signal (Tb=1)
    for NFR = 1:1:30; % Near-far ratio, in dB (Eb/N0)strong/(Eb/N0)weak
        A1 = A2*(10^((NFR/20))); % Amplitude of user 1 MSK signal
        Eb1n = (A1^2)*fsT/2; % Eb/N0 for user 1, numeric
        Eb1 = 10*log10(Eb1n); % Eb/N0 for user 1, in dB
        Na = 1; % Na = number of antennas used; must be either 1 or 2
\( \text{Eb1eff(NFR)} = 10 \log_{10}(\text{Eb1n}/(1+\text{Eb2n})) \); \% Effective Eb/N0 for user 1, Eb1/(N0+Eb2) \\
\( \text{Eb2eff(NFR)} = 10 \log_{10}(\text{Eb2n}/(1+\text{Eb1n})) \); \% Effective Eb/N0 for user 2, Eb2/(N0+Eb1) \\
fD1 = 0.01; \quad \% \text{Doppler frequency of strong (user 1) signal, relative to } T_b, \text{ i.e., } f_D = f_{\text{Doppler}} T_b \\
\phi_1 = 2 \pi \text{rand}(1,1); \quad \% \text{Carrier phase of strong signal} \\
\tau_{\text{as}} = \text{rand}(1,1)^*/f_s T; \quad \% \text{Sampling phase offset for strong signal 1, uniform between 0 and} \\
T_s/N_s = T_s/2/f_s T = 2 T/2/f_s T = 1/f_s T \\
\tau_{\text{as}} = 0; \quad \% \text{Percentage amplitude estimation error on strong signal, used to assess effect of} \\
A_e = 0; \quad \% \text{Percentage frequency estimation error on strong signal} \\
N = 400; \quad \% \text{Number of bits used, must be EVEN. Generally, } 200 < N < 800. \\
\text{if } \text{mod}(N,2) = 0; N = N + 1; \text{ end} \\
L_v = f_s T N; \quad \% \text{Length of complex transmitted vectors } v_1 \text{ and } v_2 \\
L_v = L_v + f_s T; \quad \% \text{Extend vector by one bit, f_s T samples} \\
L_v A = 3 * L_v; \quad \% \text{Allow for asynchronous reception, as follows:} \\
t = 0:1/f_s T:N-1/f_s T; \quad \% \text{Create time vector for plots} \\
\% \text{Generate MSK via OFFSET QPSK, filtered with a sinusoidal pulse.} \\
u = \sin(\pi/T_s*(0:1/f_s T:T_s-1/f_s T)+\tau_{\text{as}}^* \pi/T_s); \quad \% \text{Create sinusoidal pulse shape} \\
Nover = 32; \\
uV = \sin(\pi/2*(0:1/(Nover/2):2-1/(Nover/2))); \quad \% \text{Create "template" pulse, Nover samples/T_s, to which estimate can be compared} \\
b1 = \text{BIN01}(N,0.5); \\
b2 = \text{BIN01}(N,0.5); \% \text{Generate random binary vectors} \\
b_{\text{over1}} = \text{OverN}(b1,f_s T); \\
b_{\text{over2}} = \text{OverN}(b2,f_s T); \quad \% \text{Oversample} \\
I_1 = \text{zeros}(1,N*f_s T+f_s T); \\
Q_1 = \text{zeros}(1,N*f_s T+f_s T); \quad \% \text{Initialize } I_1 \text{ and } Q_1 \text{ components} \\
I_2 = \text{zeros}(1,N*f_s T+f_s T); \\
Q_2 = \text{zeros}(1,N*f_s T+f_s T); \quad \% \text{Initialize } I_2 \text{ and } Q_2 \text{ components} \\
\text{for } ii = 1:2:N; \quad \% \text{Loop to create MSK baseband vectors} \\
\quad I_1((ii-1)*f_s T+1:(ii+1)*f_s T) = (-2*b1(ii)+1)*u; \quad \% '0' maps to 1; '%1' maps to -1 \\
\quad Q_1((ii)*f_s T+1:(ii+2)*f_s T) = (-2*b1(ii+1)+1)*u; \\
\quad I_2((ii-1)*f_s T+1:(ii+1)*f_s T) = (-2*b2(ii)+1)*u; \\
\quad Q_2((ii)*f_s T+1:(ii+2)*f_s T) = (-2*b2(ii+1)+1)*u;
end

% Generate Doppler envelope for strong signal
Ad1c=cos(2*pi*fD1*[0:N*fsT+fsT-1]/fsT+phi1);
Ad1s=sin(2*pi*fD1*[0:N*fsT+fsT-1]/fsT+phi1);

% Generate baseband MSK signals, with Doppler envelope (so far Doppler only on strong signal)
v1=A1*(I1(1:N*fsT+fsT)-Q1(1:N*fsT+fsT));
v1=v1+j*A1*(-I1(1:N*fsT+fsT)-Q1(1:N*fsT+fsT));
v2=A2*(I2(1:N*fsT+fsT)-j*Q2(1:N*fsT+fsT));

% For burst Asynchronous, set bA=1
bA=0;
if bA == 1
    v1=[zeros(1,Lv) v1 zeros(1,Lv)];   % Lengthen signal 1 to 3*Lv for async
    tau=floor(rand(1,1)*(2*Lv+1));   % User 2 random delay, between 0 and 2*Lv samples
    v2=[zeros(1,tau) v2 zeros(1,2*Lv-tau)];   % User 2 signal, async
    n1=(randn(1,LvA)+j*randn(1,LvA))*1;   % Baseband AWGN
    rA1=v1+v2+n1;   % Baseband received signal
    rs1=rA1(Lv+1:2*Lv);   % Extract signal 1 subsequence from r
else
    v2=cshift(v2,Randinteg(1,1,fsT-1));
    n1=randn(1,Lv)+j*randn(1,Lv);
    rA1=v1+v2+n1;
    rs1=rA1;   % Extract signal 1 subsequence from r
    tau=0;
end

rAe=sqrt(sum(real(rs1).^2+imag(rs1).^2)/length(rs1));   % sqrt(rI^2 + rQ^2 ) amp estimate

I1hat=real(rs1(1:N*fsT)); Q1hat=-imag(rs1(fsT+1:N*fsT+fsT));

% Organize samples by position within each symbol--get 2*fsT samples/symbol, so 2*fsT sub-vectors
Imat=reshape(I1hat,2*fsT,N/2); Qmat=reshape(Q1hat,2*fsT,N/2);
Im=Imat'; Qm=Qmat';   % Each column of Im (or Qm) has samples in same position

% uhatI and uhatQ are each vectors of 2*fsT samples, estimate of pulse sin(pi*(t+taus)/(2*T)),
t=k*fsT, k=0,1,...2*fsT-1
uhatI=2*sum(abs(Im))/N; uhatQ=2*sum(abs(Qm))/N;
% Next need to "extract" amplitude A1 from these averaged pulse samples;
uest=(uhatI+uhatQ)/2;   % Averaging of BOTH I and Q samples together
uestS=overN(uest/A1,Nover/Ns);  % Oversample estimate for determining delay estimate tauhat 
(assumes "perfect" scaling with true A1)
uev=abs(uestS-uV);              % Compare oversampled estimate with"template" pulse uV
[du,iu]=min(uev);               % Find index of minimum difference between estimate & template
mdelay=mod(iu,Nover/Ns);        % NOTE: NOT CERTAIN OF ACCURACY of the "if" function here
if mdelay == 0;
    mdelay;
tauh=((Nover/Ns)-1)/Nover;
else
    tauh=(mod(iu,Nover/Ns)-1)/Nover; % tauh is quantized version of tauhat, in increments of 1/Nover
end

tauhat=Ts*tauh;                 % Final estimate of taus
u2=sin(pi/Ts*(0:1/fsT:Ts-1/fsT)+tauhat*pi/Ts);
A1hat=max(uest)/max(u2);

if BER1 == 1;                   % Compute user 1 Pb estimate if flag set
    % Initialize Pb's
    Ih11=zeros(1,N*fsT);     Qh11=zeros(1,N*fsT+fsT); % Initialize I1 and Q1 component estimates
    Ih12=zeros(1,N*fsT);     Qh12=zeros(1,N*fsT+fsT); % Initialize I2 and Q2 component estimates
    % Next, receiver correlation and detection for signal 1, assuming estimated carrier freq & phase, &
    perfect symbol timing
    for ir=1:2:N
        Ih11((ir-1)*fsT+1:(ir+1)*fsT)=u.*real(rs1((ir-1)*fsT+1:(ir+1)*fsT))*sqrt(2);
        Ihat11((ir+1)/2)=sign(sum(Ih11((ir-1)*fsT+1:(ir+1)*fsT)));  %I-channel estimate of signal 1, hard decision
        Ihat11((ir+1)/2)=sum(Ih11((ir-1)*fsT+1:(ir+1)*fsT));  %I-channel estimate of signal 1, soft decision
        Qh11(ir*fsT+1:(ir+2)*fsT)=-u.*imag(rs1(ir*fsT+1:(ir+2)*fsT))*sqrt(2);
        Qhat11((ir+1)/2)=sign(sum(Qh11(ir*fsT+1:(ir+2)*fsT)));%Q-channel estimate of signal 1, hard decision
        Qhat11((ir+1)/2)=sum(Qh11(ir*fsT+1:(ir+2)*fsT));% Q-channel estimate of signal 1, soft decision
    end
    bh11=LEAVE2(Ihat11,Qhat11); % Interleave soft I and Q estimates for signal 1
    bh1=(1-sign(bh11))/2;  % Hard decision estimates for signal 1
bh1H=sign(bh1); % Hard bit decisions on signal 1
bhat1=(1-bh1H)/2; % Convert +/-1 values to 0/1 values
er1=abs(b1-bhat1); % Count bit errors, and compute Pb1 estimate
Pb1=sum(er1)/N
end

% Next, IC: remodulate estimated user 1 bits and subtract from r, then %demodulate user 2, on each antenna separately
I1r1=zeros(1,Lv);  Q1r1=zeros(1,Lv);  % Initialize reconstructed I1 and %Q1 components, antenna 1
% I1r2=zeros(1,Lv);  Q1r2=zeros(1,Lv);  % Initialize reconstructed I1 and %Q1 components, antenna 2
for ic=1:2:N;  % Loop to remodulate user 1 bit estimates
  I1r1((ic-1)*fsT+1:(ic+1)*fsT) = (-2*bh1(ic)+1)*u;
  Q1r1(ic*fsT+1:(ic+2)*fsT) = (-2*bh1(ic+1)+1)*u;
  %   I1r2((ic-1)*fsT+1:(ic+1)*fsT) = (-2*bhat1A2(ic)+1)*u;
  %   Q1r2(ic*fsT+1:(ic+2)*fsT) = (-2*bhat1A2(ic+1)+1)*u;
end
%   A1h=A1+((2*BIN01(1,0.5)-1)*Ae/100)*A1; % Introduce strong-signal amplitude error of Ae
% % Reconstructed user 1 baseband MSK signal, using estimated amp & gain,
% %ant 1, with estimated Doppler
v1r1=A1hat*(I1r1(1:N*fsT+fsT)-Q1r1(1:N*fsT+fsT));
%   v1r1=v1r1+j*A1hat*(-I1r1(1:N*fsT+fsT)-Q1r1(1:N*fsT+fsT));
r1=rA1-v1r1;% IC the user 1 signal from the first-stage %received signal, ant 1
rICs1=r1;
for i2=1:2:N
  Ih21((i2-1)*fsT+1:(i2+1)*fsT) = u.*real(rICs1((i2-1)*fsT+1:(i2+1)*fsT))*sqrt(2);
  Ihat21((i2+1)/2)=sum(Ih21((i2-1)*fsT+1:(i2+1)*fsT));    % Antenna 1 I-channel estimate of signal 2, soft decision
  Qh21((i2+1)/2)=sum(Qh21((i2-1)*fsT+1:(i2+1)*fsT));        % Antenna 1 Q-channel estimate of signal 2, soft decision
end
bh21=LEAVE2(Ihat21,Qhat21); % Interleave soft I and Q estimates for signal 2, antenna 1
bhat2A1=(1-sign(bh21))/2; % Hard decision estimates for signal 2, antenna 1
bh2=bh21; % Combine soft estimates of signal 2 from both antennas, then make hard decision.

Current combining method is MRC

bh2H=sign(bh2); % Hard bit decisions on signal 2, after IC
bhat2=(1-bh2H)/2; % Convert +/-1 values to 0/1 values
er2=abs(b2-bhat2); % Count bit errors, and compute Pb2 estimate
Pb2(NFR)=sum(er2)/N;
NFR=NFR+1;
end
Pbstar2=Pbstar2+Pb2;
end
avg2=Pbstar2/300;
disp('the values of Eb2 in db are');
disp(avg2);

j=sqrt(-1);
BER1=1; % If user-1 BER computation desired, set BER1=1; **NOT QUITE ACCURATE IN SECTION AT BOTTOM**
fsT=4; % Normalized sampling rate, per bit (fs=sampling frequency, T=bit time=1 here)
T=1;Ts=2*T; % Set bit and symbol durations
Ns=2*fsT; % Ns=#samples/symbol=#samples/Ts=2*fsT
Ntrials=1;
Pbstar3=zeros(1,30);
for i=1:1:300
  j=sqrt(-1);
  fsT=4; % Normalized sampling rate, per bit (fs=sampling frequency,
  % T=bit time =1 here)
  Eb2=4; % Energy per bit to spectral noise density ratio% (Eb/N0) for user 2, in dB
  Eb2n=10^(Eb2/10); % Eb/N0 for user 2, numeric
  A2=sqrt(2*Eb2n/fsT);% Amplitude of user 2 MSK signal (Tb=1)
  for NFR=1:1:30; % Near-far ratio, in dB (Eb/N0)strong/(Eb/N0)weak
    A1=A2*(10^(NFR/20));% Amplitude of user 1 MSK signal
    Eb1n=(A1^2)*fsT/2; % Eb/N0 for user 1, numeric
    Eb1=10*log10(Eb1n); % Eb/N0 for user 1, in dB
    Na=1; % Na = number of antennas used; must be either 1 or 2
    Eb1eff(NFR)=10*log10(Eb1n/(1+Eb2n));% Effective Eb/N0 for user 1,Eb1/(N0+Eb2)
\(Eb2eff(NFR)=10\log_{10}(Eb2n/(1+Eb1n))\); \(\text{Effective } Eb/N0 \text{ for user 2, } Eb2/(N0+Eb1)\)

\(fD1=0.01\); \(\text{Doppler frequency of strong (user 1) signal, relative to } Tb, \text{ i.e., } fD=fDoppler*Tb\)

\(phi1=2*pi*rand(1,1)\); \(\text{Carrier phase of strong signal}\)

\(taus=rand(1,1)^*/fsT\); \(\text{Sampling phase offset for strong signal 1, uniform between 0 and}\)

\(Ts/Ns=Ts/2/fsT=2*T/2/fsT=1/fsT\)

\(taus=0;\)

\(Ae=0;\) \(\text{Percentage amplitude estimation error on strong signal, used to assess effect of}\)

\(f1e=0;\) \(\text{Percentage frequency estimation error on strong signal}\)

\(N=400;\) \(\text{Number of bits used, must be EVEN. Generally, } 200<N<800.\)

\(\text{if } \text{mod}(N,2) \neq 0; \text{N=N+1; end}\)

\(Lv=fsT*N;\) \(\text{Length of complex transmitted vectors v1 and v2}\)

\(Lv=Lv+fsT;\) \(\text{Extend vector by one bit, } fsT \text{ samples}\)

\(LvA=3*Lv;\) \(\text{Allow for asynchronous reception, as follows:}\)

\(t=0:1/fsT:N-1/fsT;\) \(\text{Create time vector for plots}\)

\(\text{Generate MSK via OFFSET QPSK, filtered with a sinusoidal pulse.}\)

\(u=sin(pi/Ts*(0:1/fsT:Ts-1/fsT)+taus*pi/Ts);\) \(\text{Create sinusoidal pulse shape}\)

\(Nover=32;\)

\(uV=sin(pi/2*(0:1/(Nover/2):2-1/(Nover/2)));\) \(\text{Create "template" pulse, } Nover \text{ samples/Ts, to which estimate can be compared}\)

\(b1=BIN01(N,0.5);\)

\(b2=BIN01(N,0.5);\) \(\text{Generate random binary vectors}\)

\(bover1=OverN(b1,fsT);\)

\(bover2=OverN(b2,fsT);\) \(\text{Oversample}\)

\(I1=zeros(1,N*fsT+fsT);\)

\(Q1=zeros(1,N*fsT+fsT);\) \(\text{Initialize I1 and Q1 components}\)

\(I2=zeros(1,N*fsT+fsT);\)

\(Q2=zeros(1,N*fsT+fsT);\) \(\text{Initialize I2 and Q2 components}\)

\(\text{for } ii=1:2:N;\) \(\text{Loop to create MSK baseband vectors}\)

\(I1((ii-1)*fsT+1:(ii+1)*fsT)=(-2*b1(ii)+1)*u;\) \(\text{'}0' \text{ maps to 1; } \text{'1' maps to -1}\)

\(Q1((ii-1)*fsT+1:(ii+2)*fsT)=(-2*b1(ii+1)+1)*u;\)

\(I2((ii-1)*fsT+1:(ii+1)*fsT)=(-2*b2(ii)+1)*u;\)

\(Q2((ii-1)*fsT+1:(ii+2)*fsT)=(-2*b2(ii+1)+1)*u;\)

\(\text{end}\)
% Generate Doppler envelope for strong signal
A1c=cos(2*pi*fD1*[0:N*fsT+fsT-1]/fsT+phi1);
A1s=sin(2*pi*fD1*[0:N*fsT+fsT-1]/fsT+phi1);

% Generate baseband MSK signals, with Doppler envelope (so far Doppler only on strong signal)
v1=A1*(I1(1:N*fsT+fsT)-Q1(1:N*fsT+fsT));
v1=v1+j*A1*(-I1(1:N*fsT+fsT)-Q1(1:N*fsT+fsT));
v2=A2*(I2(1:N*fsT+fsT)+j*Q2(1:N*fsT+fsT));

% For burst Asynchronous, set bA=1
bA=0;
if bA == 1
    v1=[zeros(1,Lv) v1 zeros(1,Lv)]; % Lengthen signal 1 to 3*Lv for async
    tau=floor(rand(1,1)*(2*Lv+1)); % User 2 random delay, between 0 and 2*Lv samples
    v2=[zeros(1,tau) v2 zeros(1,2*Lv-tau)]; % User 2 signal, async
    n1=(randn(1,LvA)+j*randn(1,LvA))*1; % Baseband AWGN
    rA1=v1+v2+n1; % Baseband received signal
    rs1=rA1(Lv+1:2*Lv); % Extract signal 1 subsequence from r
else
    v2=cshift(v2,randinteg(1,1,fsT-1));
    n1=randn(1,Lv)+j*randn(1,Lv);
    rA1=v1+v2+n1;
    rs1=rA1; % Extract signal 1 subsequence from r
    tau=0;
end
rAe=sqrt(sum(real(rs1).^2+imag(rs1).^2)/length(rs1)); % sqrt(rI^2 + rQ^2) amp estimate

I1hat=real(rs1(1:N*fsT)); Q1hat=-imag(rs1(fsT+1:N*fsT+fsT));
% Organize samples by position within each symbol--get 2*fsT samples/symbol, so 2*fsT sub-vectors
Imat=reshape(I1hat,2*fsT,N/2); Qmat=reshape(Q1hat,2*fsT,N/2);
Im=Imat'; Qm=Qmat'; % Each column of Im (or Qm) has samples in same position
% uhatI and uhatQ are each vectors of 2*fsT samples, estimate of pulse sin(pi*(t+taus)/(2*T)), t=k*fsT, k=0,1,...,2*fsT-1
uhatI=2*sum(abs(Im))/N; uhatQ=2*sum(abs(Qm))/N;
% Next need to "extract" amplitude A1 from these averaged pulse samples;
uest=(uhatI+uhatQ)/2; % Averaging of BOTH I and Q samples together
uestS=overN(uest/A1,Nover/Ns); % Oversample estimate for determining delay estimate tauhat
(assumes "perfect" scaling with true A1)
uev = abs(uestS-uV);              % Compare oversampled estimate with "template" pulse uV
[du,iu] = min(uev);               % Find index of minimum difference between estimate & template
mdelay = mod(iu,Nover/Ns);        % NOTE: NOT CERTAIN OF ACCURACY of the "if" function here
if mdelay == 0;
    mdelay;
    tauh = ((Nover/Ns)-1)/Nover;
else
    tauh = (mod(iu,Nover/Ns)-1)/Nover; % tauh is quantized version of tauhat, in increments of 1/Nover
end
tauhat = Ts*tauh;                 % Final estimate of taus
u2 = sin(pi/Ts*(0:1/fsT:Ts-1/fsT)+tauhat*pi/Ts);
A1hat = max(uest)/max(u2);

if BER1 == 1; % Compute user 1 Pb estimate if flag set
    Pb1=0; Pb2=0; % Initialize Pb's
    Ih11 = zeros(1,N*fsT); Qh11 = zeros(1,N*fsT+fsT); % Initialize I1 and Q1 component estimates
    Ih12 = zeros(1,N*fsT); Qh12 = zeros(1,N*fsT+fsT); % Initialize I2 and Q2 component estimates
    % Next, receiver correlation and detection for signal 1, assuming estimated carrier freq & phase, &
    % perfect symbol timing
    for ir=1:2:N
        Ih11((ir-1)*fsT+1:(ir+1)*fsT)=u.*real(rs1((ir-1)*fsT+1:(ir+1)*fsT))*sqrt(2);
        Ihat11((ir+1)/2)=sign(sum(Ih11((ir-1)*fsT+1:(ir+1)*fsT))); %I-channel estimate of signal 1, hard decision
        Qh11((ir+1)/2)=sign(sum(Qh11((ir-1)*fsT+1:(ir+1)*fsT))); %Q-channel estimate of signal 1, hard decision
        Qhat11((ir+1)/2)=sign(sum(Qh11((ir+1)*fsT+1:(ir+2)*fsT))); %Q-channel estimate of signal 1, soft decision
    end
    bh11 = LEAVE2(Ihat11,Qhat11); % Interleave soft I and Q estimates for signal 1
    bh1 = (1-sign(bh11))/2; % Hard decision estimates for signal 1
    bh1H = sign(bh1); % Hard bit decisions on signal 1
    bhat1 = (1-bh1H)/2; % Convert +/-1 values to 0/1 values
er1=abs(b1-bhat1); % Count bit errors, and compute Pb1 estimate
Pb1=sum(er1)/N
end

% Next, IC: remodulate estimated user 1 bits and subtract from r, then %demodulate user 2, on each antenna separately
I1r1=zeros(1,Lv); Q1r1=zeros(1,Lv); % Initialize reconstructed I1 and Q1 components, antenna 1
I1r2=zeros(1,Lv); Q1r2=zeros(1,Lv); % Initialize reconstructed I1 and Q1 components, antenna 2
for ic=1:2:N; % Loop to remodulate user 1 bit estimates
    I1r1((ic-1)*fsT+1:(ic+1)*fsT) = (-2*bh1(ic)+1)*u;
    Q1r1(ic*fsT+1:(ic+2)*fsT) = (-2*bh1(ic+1)+1)*u;
end
% A1h=A1+((2*BIN01(1,0.5)-1)*Ae/100)*A1; % Introduce strong-signal amplitude error of Ae
% Reconstructed user 1 baseband MSK signal, using estimated amp & gain, %ant 1, with estimated Doppler
v1r1=A1hat*(I1r1(1:N*fsT+fsT)-Q1r1(1:N*fsT+fsT));
    v1r1=v1r1+j*A1hat*(-I1r1(1:N*fsT+fsT)-Q1r1(1:N*fsT+fsT));
    r1=rA1-v1r1;% IC the user 1 signal from the first-stage %received signal, ant 1
    rICs1=r1;
for i2=1:2:N
    Ih21((i2-1)*fsT+1:(i2+1)*fsT) = u.*real(rICs1((i2-1)*fsT+1:(i2+1)*fsT))*sqrt(2);
    Ihat21((i2+1)/2)=sum(Ih21((i2-1)*fsT+1:(i2+1)*fsT)); % Antenna 1 I-channel estimate of signal 2, soft decision
    Qh21((i2+1)*fsT+1:(i2+2)*fsT))=sum(Qh21((i2+1)*fsT+1:(i2+2)*fsT)); % Antenna 1 Q-channel estimate of signal 2, soft decision
end
bh2=LEAVE2(Ihat21,Qhat21); % Interleave soft I and Q estimates for signal 2, antenna 1
bhat2A1=(1-sign(bh21))/2; % Hard decision estimates for signal 2, antenna 1
bh2=bh21; % Combine soft estimates of signal 2 from both antennas, then make hard decision.
Current combining method is MRC
bh2H=sign(bh2);  \text{ % Hard bit decisions on signal 2, after IC}

bhat2=(1-bh2H)/2;  \text{ % Convert +/-1 values to 0/1 values}

er2=abs(b2-bhat2);  \text{ % Count bit errors, and compute Pb2 estimate}

Pb2(NFR)=sum(er2)/N;

NFR=NFR+1;

end

Pbstar3=Pbstar3+Pb2;

end

avg3=Pbstar3/300;
disp('the values of Eb2 in db are');
disp(avg3);

j=sqrt(-1);

BER1=1;  \text{ % If user-1 BER computation desired, set BER1=1; **NOT QUITE ACCURATE IN SECTIONS AT BOTTOM***}

fsT=4;  \text{ % Normalized sampling rate, per bit (fs=sampling frequency, T=bit time=1 here)}

T=1;Ts=2*T;  \text{ % Set bit and symbol durations}

Ns=2*fsT;  \text{ % Ns=#samples/symbol=#samples/Ts=2*fsT}

Ntrials=1;

Pbstar4=zeros(1,30);

for i=1:1:300

j=sqrt(-1);

fsT=4;  \text{ % Normalized sampling rate, per bit (fs=sampling frequency, T=bit time=1 here)}

Eb2=4;  \text{ % Energy per bit to spectral noise density ratio (Eb/N0) for user 2, in dB}

Eb2n=10^(Eb2/10);  \text{ % Eb/N0 for user 2, numeric}

A2=sqrt(2*Eb2n/fsT);  \text{ % Amplitude of user 2 MSK signal (Tb=1)}

for NFR=1:1:30;  \text{ % Near-far ratio, in dB (Eb/N0)strong/(Eb/N0)weak}

A1=A2*(10^((NFR/20)));  \text{ % Amplitude of user 1 MSK signal}

Eb1n=(A1^2)*fsT/2;  \text{ % Eb/N0 for user 1, numeric}

Eb1=10*log10(Eb1n);  \text{ % Eb/N0 for user 1, in dB}

Na=1;  \text{ % Na = number of antennas used; must be either 1 or 2}

Eb1eff(NFR)=10*log10(Eb1n/(1+Eb2n));  \text{ % Effective Eb/N0 for user 1, Eb1/(N0+Eb2)}

Eb2eff(NFR)=10*log10(Eb2n/(1+Eb1n));  \text{ % Effective Eb/N0 for user 2, Eb2/(N0+Eb1)}

fD1=0.01;  \text{ % Doppler frequency of strong (user 1) signal, %relative to Tb, i.e., fD=fDoppler*Tb}
phi1=2*pi*rand(1,1); % Carrier phase of strong signal
taus=rand(1,1)*1/fsT; % Sampling phase offset for strong signal 1, uniform between 0 and
Ts/Ns=Ts/2/fsT=2*T/2/fsT=1/fsT
taus=0;
Ae=0; % Percentage amplitude estimation error on strong signal, used to assess effect of
imperfect amp estimation on IC
f1e=0; % Percentage frequency estimation error on strong signal
N=400; % Number of bits used, must be EVEN. Generally,
200<N<800.
if mod(N,2) ~= 0; N=N+1; end
Lv=fsT*N; % Length of complex transmitted vectors v1 and v2
Lv=Lv+fsT; % Extend vector by one bit, fsT samples
LvA=3*Lv; % Allow for asynchronous reception, as follows:
t=0:1/fsT:N-1/fsT; % Create time vector for plots
% Generate MSK via OFFSET QPSK, filtered with a sinusoidal pulse.
u=sin(pi/Ts*(0:1/fsT:Ts-1/fsT)+taus*pi/Ts); % Create sinusoidal pulse shape
Nover=32;
uV=sin(pi/2*(0:1/(Nover/2):2-1/(Nover/2))); % Create "template" pulse, Nover samples/Ts, to
which estimate can be compared
b1=BIN01(N,0.5);
b2=BIN01(N,0.5); % Generate random binary vectors
bover1=OverN(b1,fsT);
bover2=OverN(b2,fsT); % Oversample
I1=zeros(1,N*fsT+fsT);
Q1=zeros(1,N*fsT+fsT); % Initialize I1 and Q1 components
I2=zeros(1,N*fsT+fsT);
Q2=zeros(1,N*fsT+fsT); % Initialize I2 and Q2 components
for ii=1:2:N; % Loop to create MSK baseband vectors
I1((ii-1)*fsT+1:(ii+1)*fsT) = (-2*b1(ii)+1)*u; % '0' maps to 1; '%1' maps to -1
Q1((ii-1)*fsT+1:(ii+2)*fsT) = (-2*b1(ii)+1)*u;
I2((ii-1)*fsT+1:(ii+1)*fsT) = (-2*b2(ii)+1)*u;
Q2((ii-1)*fsT+1:(ii+2)*fsT) = (-2*b2(ii)+1)*u;
end
% Generate Doppler envelope for strong signal
Ad1c=cos(2*pi*fD1*[0:N*fsT+fsT-1]/fsT+phi1);
Adls = \text{sin}(2\pi fD1*[0:N*fsT+fsT-1]/fsT+phi1); \\
% Generate baseband MSK signals, with Doppler envelope (so far Doppler only on strong signal) \\
v1=A1*(I1(1:N*fsT+fsT)-Q1(1:N*fsT+fsT)); \\
v1=v1+j*A1*(-I1(1:N*fsT+fsT)-Q1(1:N*fsT+fsT)); \\
v2=A2*(I2(1:N*fsT+fsT)-j*Q2(1:N*fsT+fsT)); \\
% For burst Asynchronous, set bA=1 \\
bA=0; \\
if bA == 1 \\
  v1=[zeros(1,Lv) v1 zeros(1,Lv)]; % Lengthen signal 1 to 3*Lv for async \\
  tau=floor(rand(1,1)*(2*Lv+1)); % User 2 random delay, between 0 and 2*Lv samples \\
  v2=[zeros(1,tau) v2 zeros(1,2*Lv-tau)]; % User 2 signal, async \\
  n1=(randn(1,LvA)+j*randn(1,LvA))*1; % Baseband AWGN \\
  rA1=v1+v2+n1; % Baseband received signal \\
  rs1=rA1(Lv+1:2*Lv); % Extract signal 1 subsequence from r \\
else \\
  v2=cshift(v2,Randinteg(1,1,fsT-1)); \\
  n1=randn(1,Lv)+j*randn(1,Lv); \\
  rA1=v1+v2+n1; \\
  rs1=rA1; % Extract signal 1 subsequence from r \\
  tau=0; \\
end \\
rAe=sqrt(sum(real(rs1).^2+imag(rs1).^2)/length(rs1)); % sqrt(rI^2 + rQ^2) amp estimate \\
I1hat=real(rs1(1:N*fsT)); Q1hat=-imag(rs1(fsT+1:N*fsT+fsT)); \\
% Organize samples by position within each symbol--get 2*fsT samples/symbol, so 2*fsT sub-vectors \\
Imat=reshape(I1hat,2*fsT,N/2); Qmat=reshape(Q1hat,2*fsT,N/2); \\
Im=Imat'; Qm=Qmat'; % Each column of Im (or Qm) has samples in same position \\
% uhatl and uhatQ are each vectors of 2*fsT samples, estimate of pulse sin(pi*(t+taus)/(2*T)), \\
t=k*fsT, k=0,1,...2*fsT-1 \\
uhatl=2*sum(abs(Im))/N; uhatQ=2*sum(abs(Qm))/N; \\
% Next need to "extract" amplitude A1 from these averaged pulse samples; \\
uest=(uhatl+uhatQ)/2; % Averaging of BOTH I and Q samples together \\
uestS=overN(uest/A1,Nover/Ns); % Oversample estimate for determining delay estimate tauhat \\
(assumes "perfect" scaling with true A1) \\
uev=abs(uestS-uV); % Compare oversampled estimate with"template" pulse uV \\
[du,iu]=min(uev); % Find index of minimum difference between estimate & template
mdelay = mod(iu, Nover/Ns);  % NOTE: NOT CERTAIN OF ACCURACY of the "if" function here
if mdelay == 0;
    mdelay;
tauh = ((Nover/Ns)-1)/Nover;
else
    tauh = (mod(iu, Nover/Ns)-1)/Nover;  % tauh is quantized version of tauhat, in increments of 1/Nover
end

tauhat = Ts*tauh;  % Final estimate of taus

u2 = sin(pi/Ts*(0:1/fsT:Ts-1/fsT)+tauhat*pi/Ts);
A1hat = max(uest)/max(u2);

if BER1 == 1;  % Compute user 1 Pb estimate if flag set
    % Initialize Pb's
    Ih11 = zeros(1, N*fsT);
    Qh11 = zeros(1, N*fsT+fsT);  % Initialize I1 and Q1 component estimates
    Ih12 = zeros(1, N*fsT);
    Qh12 = zeros(1, N*fsT+fsT);  % Initialize I2 and Q2 component estimates
    % Next, receiver correlation and detection for signal 1, assuming estimated carrier freq & phase, &
    % perfect symbol timing
    for ir=1:2:N
        Ih11((ir-1)*fsT+1:(ir+1)*fsT)=u.*real(rs1((ir-1)*fsT+1:(ir+1)*fsT))*sqrt(2);
        Ihat11((ir+1)/2)=sign(sum(Ih11((ir-1)*fsT+1:(ir+1)*fsT)));  %I-channel estimate of signal 1, hard decision
        Ihat11((ir+1)/2)=sum(Ih11((ir-1)*fsT+1:(ir+1)*fsT));  %I-channel estimate of signal 1, soft decision
        Qh11(ir*fsT+1:(ir+2)*fsT)=-u.*imag(rs1(ir*fsT+1:(ir+2)*fsT))*sqrt(2);
        Qhat11((ir+1)/2)=sign(sum(Qh11(ir*fsT+1:(ir+2)*fsT)));  %Q-channel estimate of signal 1, hard decision
        Qhat11((ir+1)/2)=sum(Qh11(ir*fsT+1:(ir+2)*fsT));  %Q-channel estimate of signal 1, soft decision
    end
    bh11=LEAVE2(Ihat11,Qhat11);  % Interleave soft I and Q estimates for signal 1
    bh1=(1-sign(bh11))/2;  % Hard decision estimates for signal 1
    bh1H=sign(bh1);  % Hard bit decisions on signal 1
    bhat1=(1-bh1H)/2;  % Convert +/-1 values to 0/1 values
    er1=abs(b1-bhat1);  % Count bit errors, and compute Pb1 estimate
    Pb1=sum(er1)/N
% Next, IC: remodulate estimated user 1 bits and subtract from r, then %demodulate user 2, on each antenna separately

I1r1=zeros(1,Lv);  Q1r1=zeros(1,Lv);  % Initialize reconstructed I1 and %Q1 components, antenna 1
I1r2=zeros(1,Lv);  Q1r2=zeros(1,Lv);  % Initialize reconstructed I1 and %Q1 components, antenna 2
for ic=1:2:N;               % Loop to remodulate user 1 bit estimates
    I1r1((ic-1)*fsT+1:(ic+1)*fsT) = (-2*bh1(ic)+1)*u;
    Q1r1(ic*fsT+1:(ic+2)*fsT) = (-2*bh1(ic+1)+1)*u;
%         I1r2((ic-1)*fsT+1:(ic+1)*fsT) = (-2*bhat1A2(ic)+1)*u;
%         Q1r2(ic*fsT+1:(ic+2)*fsT) = (-2*bhat1A2(ic+1)+1)*u;
end
%         A1h=A1+((2*BIN01(1,0.5)-1)*Ae/100)*A1; % Introduce strong-signal amplitude error of Ae
%% Reconstructed user 1 baseband MSK signal, using estimated amp & gain,
%ant 1, with estimated Doppler
v1r1=A1hat*(I1r1(1:N*fsT+fsT)-Q1r1(1:N*fsT+fsT));
V1r1=v1r1+j*A1hat*(-I1r1(1:N*fsT+fsT)-Q1r1(1:N*fsT+fsT));
r1=rA1-v1r1;% IC the user 1 signal from the first-stage %received signal, ant 1
rICs1=r1;
for i2=1:2:N
    Ih21((i2-1)*fsT+1:(i2+1)*fsT) = u.*real(rICs1((i2-1)*fsT+1:(i2+1)*fsT))*sqrt(2);
    Ihat21((i2+1)/2)=sum(Ih21((i2-1)*fsT+1:(i2+1)*fsT));    % Antenna 1 I-channel estimate of signal 2, soft decision
    Qh21((i2+1)/2) = -u.*imag(rICs1(i2*fsT+1:(i2+2)*fsT))*sqrt(2);
    Qhat21((i2+1)/2)=sum(Qh21(i2*fsT+1:(i2+2)*fsT));        % Antenna 1 Q-channel estimate of signal 2, soft decision
end
bh21=LEAVE2(Ihat21,Qhat21); % Interleave soft I and Q estimates for signal 2, antenna 1
bhat2A1=(1-sign(bh21))/2;   % Hard decision estimates for signal 2, antenna 1
bh2=bh21; % Combine soft estimates of signal 2 from both antennas, then make hard decision.
Current combining method is MRC
bhat2H=sign(bh2); % Hard bit decisions on signal 2, after IC
bhat2=(1-bh2H)/2; % Convert +/-1 values to 0/1 values
er2 = abs(b2-bhat2);  % Count bit errors, and compute Pb2 estimate
Pb2(NFR)=sum(er2)/N;
NFR=NFR+1;
end
Pbstar4=Pbstar4+Pb2;
end
avg4=Pbstar4/300;
disp('the values of Eb2 in db are');
disp(avg4);

j=sqrt(-1);
BER1=1;              % If user-1 BER computation desired, set BER1=1; **NOT QUITE ACCURATE IN SECTION AT BOTTOM***
fsT=4;     % Normalized sampling rate, per bit (fs=sampling frequency, T=bit time=1 here)
T=1;Ts=2*T;          % Set bit and symbol durations
Ns=2*fsT;       % Ns=#samples/symbol=#samples/Ts=2*fsT
Ntrials=1;
Pbstar5=zeros(1,30);
for i=1:1:300
j=sqrt(-1);
fsT=4;    % Normalized sampling rate, per bit (fs=sampling frequency, T=bit time=1 here)
% T=bit time =1 here)
Eb2=6;            % Energy per bit to spectral noise density ratio% (Eb/N0) for user 2, in dB
Eb2n=10^(Eb2/10);  % Eb/N0 for user 2, numeric
A2=sqrt(2*Eb2n/fsT);% Amplitude of user 2 MSK signal (Tb=1)
for NFR=1:1:30;             % Near-far ratio, in dB (Eb/N0)strong/(Eb/N0)weak
A1=A2*(10^(NFR/20));% Amplitude of user 1 MSK signal
Eb1n=(A1^2)*fsT/2;  % Eb/N0 for user 1, numeric
Eb1=10*log10(Eb1n); % Eb/N0 for user 1, in dB
Na=1;               % Na = number of antennas used; must be either 1 or 2
Eb1eff(NFR)=10*log10(Eb1n/(1+Eb2n));% Effective Eb/N0 for user 1,Eb1/(N0+Eb2)
Eb2eff(NFR)=10*log10(Eb2n/(1+Eb1n)); % Effective Eb/N0 for user 2,Eb2/(N0+Eb1)
fD1=0.01;           % Doppler frequency of strong (user 1) signal,%relative to Tb, i.e., fD=fDoppler*Tb
phi1=2*pi*rand(1,1); % Carrier phase of strong signal
taus=rand(1,1)*1/fsT;% Sampling phase offset for strong signal 1, uniform between 0 and
Ts/Ns=Ts/2/fsT=2*T/2/fsT=1/fsT

Ae=0; % Percentage amplitude estimation error on strong signal, used to assess effect of
imperfect amp estimation on IC

f1e=0; % Percentage frequency estimation error on strong signal

N=400; % Number of bits used, must be EVEN. Generally, 200<N<800.

if mod(N,2) ~= 0; N=N+1; end

Lv=fsT*N; % Length of complex transmitted vectors v1 and v2
Lv=Lv+fsT; % Extend vector by one bit, fsT samples
LvA=3*Lv; % Allow for asynchronous reception, as follows:

t=0:1/fsT:N-1/fsT;% Create time vector for plots

% Generate MSK via OFFSET QPSK, filtered with a sinusoidal pulse.
u=sin(pi/Ts*(0:1/fsT:Ts-1/fsT)+taus*pi/Ts); % Create sinusoidal pulse shape
Nover=32;
uV=sin(pi/2*(0:1/(Nover/2):2-1/(Nover/2))); % Create "template" pulse, Nover samples/Ts, to
which estimate can be compared

b1=BIN01(N,0.5);
b2=BIN01(N,0.5); % Generate random binary vectors
bover1=OverN(b1,fsT);
bover2=OverN(b2,fsT); % Oversample
I1=zeros(1,N*fsT+fsT);
Q1=zeros(1,N*fsT+fsT); % Initialize I1 and Q1 components
I2=zeros(1,N*fsT+fsT);
Q2=zeros(1,N*fsT+fsT); % Initialize I2 and Q2 components
for ii=1:2:N; % Loop to create MSK baseband vectors
    I1((ii-1)*fsT+1:(ii+1)*fsT) = (-2*b1(ii)+1)*u; % '0' maps to 1; % '1' maps to -1
    Q1((ii-1)*fsT+1:(ii+1)*fsT) = (-2*b1(ii+1)+1)*u;
    I2((ii-1)*fsT+1:(ii+1)*fsT) = (-2*b2(ii)+1)*u;
    Q2((ii-1)*fsT+1:(ii+1)*fsT) = (-2*b2(ii+1)+1)*u;
end

% Generate Doppler envelope for strong signal
Ad1c=cos(2*pi*fD1*[0:N*fsT+fsT-1]/fsT+phi1);
Ad1s=sin(2*pi*fD1*[0:N*fsT+fsT-1]/fsT+phi1);
% Generate baseband MSK signals, with Doppler envelope (so far Doppler only on strong signal)
v1=A1*(I1(1:N*fsT+fsT)-Q1(1:N*fsT+fsT));
v1=v1+j*A1*(-I1(1:N*fsT+fsT)-Q1(1:N*fsT+fsT));
v2=A2*(I2(1:N*fsT+fsT)-j*Q2(1:N*fsT+fsT));

% For burst Asynchronous, set bA=1
bA=0;
if bA == 1
    v1=[zeros(1,Lv) v1 zeros(1,Lv)]; % Lengthen signal 1 to 3*Lv for async
    tau=floor(rand(1,1)*(2*Lv+1)); % User 2 random delay, between 0 and 2*Lv samples
    v2=[zeros(1,tau) v2 zeros(1,2*Lv-tau)]; % User 2 signal, async
    n1=(randn(1,LvA)+j*randn(1,LvA))*1; % Baseband AWGN
    rA1=v1+v2+n1; % Baseband received signal
    rs1=rA1(Lv+1:2*Lv); % Extract signal 1 subsequence from r
else
    v2=cshift(v2,Randinteg(1,1,fsT-1));
    n1=randn(1,Lv)+j*randn(1,Lv);
    rA1=v1+v2+n1;
    rs1=rA1; % Extract signal 1 subsequence from r
    tau=0;
end
rAe=sqrt(sum(real(rs1).^2+imag(rs1).^2)/length(rs1)); % sqrt(rI^2 + rQ^2) amp estimate

I1hat=real(rs1(1:N*fsT)); Q1hat=-imag(rs1(fsT+1:N*fsT+fsT));
% Organize samples by position within each symbol--get 2*fsT samples/symbol, so 2*fsT sub-vectors
Imat=reshape(I1hat,2*fsT,N/2); Qmat=reshape(Q1hat,2*fsT,N/2);
Im=Imat'; Qm=Qmat'; % Each column of Im (or Qm) has samples in same position
% uhat_I and uhat_Q are each vectors of 2*fsT samples, estimate of pulse sin(pi*(t+taus)/(2*T)),
t=k*fsT, k=0,1,...2*fsT-1
uhat_I=2*sum(abs(Im))/N; uhat_Q=2*sum(abs(Qm))/N;
% Next need to "extract" amplitude A1 from these averaged pulse samples;
uest=(uhat_I+uhat_Q)/2; % Averaging of BOTH I and Q samples together
uestS=overN(uest/A1,Nover/Ns); % Oversample estimate for determining delay estimate tauhat
(assumes "perfect" scaling with true A1)
uev=abs(uestS-uV); % Compare oversampled estimate with "template" pulse uV
[du,ui]=min(uev); % Find index of minimum difference between estimate & template
mdelay=mod(ui,Nover/Ns); % NOTE: NOT CERTAIN OF ACCURACY of the "if" function here
if mdelay == 0;
    mdelay;
    tauh=((Nover/Ns)-1)/Nover;
else
    tauh=(mod(iu,Nover/Ns)-1)/Nover; % tauh is quantized version of tauhat, in increments of 1/Nover
end

    tauhat=Ts*tauh; % Final estimate of taus
u2=sin(pi/Ts*(0:1/fsT:Ts-1/fsT)+tauhat*pi/Ts);
A1hat=max(uest)/max(u2);

if BER1 == 1; % Compute user 1 Pb estimate if flag set
    % Initialize Pb's
    Ih11=zeros(1,N*fsT); Qh11=zeros(1,N*fsT+fsT); % Initialize I1 and Q1 component estimates
    Ih12=zeros(1,N*fsT); Qh12=zeros(1,N*fsT+fsT); % Initialize I2 and Q2 component estimates
    % Next, receiver correlation and detection for signal 1, assuming estimated carrier freq & phase, &
    % perfect symbol timing
    for ir=1:2:N
        Ih11((ir-1)*fsT+1:(ir+1)*fsT)=u.*real(rs1((ir-1)*fsT+1:(ir+1)*fsT))*sqrt(2);
        Ihat11((ir+1)/2)=sign(sum(Ih11((ir-1)*fsT+1:(ir+1)*fsT)));
        % I-channel estimate of signal 1, hard decision
        Ihat11((ir+1)/2)=sum(Ih11((ir-1)*fsT+1:(ir+1)*fsT)); % I-channel estimate of signal 1, soft decision
        Qh11(ir*fsT+1:(ir+2)*fsT)=-u.*imag(rs1(ir*fsT+1:(ir+2)*fsT))*sqrt(2);
        Qhat11((ir+1)/2)=sign(sum(Qh11(ir*fsT+1:(ir+2)*fsT)));
        % Q-channel estimate of signal 1, hard decision
        Qhat11((ir+1)/2)=sum(Qh11(ir*fsT+1:(ir+2)*fsT)); % Q-channel estimate of signal 1, soft decision
    end
    bh11=LEAVE2(Ihat11,Qhat11); % Interleave soft I and Q estimates for signal 1
    bh1=(1-sign(bh11))/2; % Hard decision estimates for signal 1
    bh1H=sign(bh1); % Hard bit decisions on signal 1
    bhat1=(1-bh1H)/2; % Convert +/-1 values to 0/1 values
    er1=abs(b1-bhat1); % Count bit errors, and compute Pb1 estimate
    Pb1=sum(er1)/N
end
% Next, IC: remodulate estimated user 1 bits and subtract from r, then %demodulate user 2, on each antenna separately

I1r1=zeros(1,Lv); Q1r1=zeros(1,Lv); % Initialize reconstructed I1 and %Q1 components, antenna 1
I1r2=zeros(1,Lv); Q1r2=zeros(1,Lv); % Initialize reconstructed I1 and %Q1 components, antenna 2

for ic=1:2:N; % Loop to remodulate user 1 bit estimates
    I1r1((ic-1)*fsT+1:(ic+1)*fsT) = (-2*bh1(ic)+1)*u;
    Q1r1((ic+2)*fsT) = (-2*bh1(ic+1)+1)*u;
    % I1r2((ic-1)*fsT+1:(ic+1)*fsT) = (-2*bhat1A2(ic)+1)*u;
    % Q1r2((ic+2)*fsT) = (-2*bhat1A2(ic+1)+1)*u;
end

% A1h=A1+(2*BIN01(1,0,5)-1)*Ae/100)*A1; % Introduce strong-signal amplitude error of Ae
% Reconstructed user 1 baseband MSK signal, using estimated amp & gain,
% ant 1, with estimated Doppler
v1r1=A1hat*(I1r1(1:N*fsT+fsT)-Q1r1(1:N*fsT+fsT));

for i2=1:2:N
    Ih21((i2-1)*fsT+1:(i2+1)*fsT) = u.*real(rICs1((i2-1)*fsT+1:(i2+1)*fsT))*sqrt(2);
    Ihat21((i2+1)/2)=sum(Ih21((i2-1)*fsT+1:(i2+1)*fsT)); % Antenna 1 I-channel estimate of signal
    Qh21(i2*fsT+1:(i2+2)*fsT) = -u.*imag(rICs1(i2*fsT+1:(i2+2)*fsT))*sqrt(2);
    Qhat21((i2+1)/2)=sum(Qh21(i2*fsT+1:(i2+2)*fsT)); % Antenna 1 Q-channel estimate of signal
end
bh21=LEAVE2(Ihat21,Qhat21); % Interleave soft I and Q estimates for signal 2, antenna 1
bhat2A1=(1-sign(bh21))/2; % Hard decision estimates for signal 2, antenna 1

bh2=bh21; % Combine soft estimates of signal 2 from both antennas, then make hard decision.
Current combining method is MRC
bh2H=sign(bh2); % Hard bit decisions on signal 2, after IC
bhat2=(1-bh2H)/2; % Convert +/-1 values to 0/1 values
er2=abs(b2-bhat2); % Count bit errors, and compute Pb2 estimate
Pb2(NFR)=sum(er2)/N;
NFR=NFR+1;
end
Pbstar5=Pbstar5+Pb2;
end
avg5=Pbstar5/300;
disp('the values of Eb2 in db are');
disp(avg5);

j=sqrt(-1);
BER1=1;              % If user-1 BER computation desired, set BER1=1; **NOT QUITE ACCURATE IN
SECTION AT BOTTOM***
fsT=4;          % Normalized sampling rate, per bit (fs=sampling frequency, T=bit
time=1 here)
T=1;Ts=2*T;            % Set bit and symbol durations
Ns=2*fsT;            % Ns=#samples/symbol=#samples/Ts=2*fsT
Ntrials=1;
Pb6=zeros(1,30);
for i=1:1:300
j=sqrt(-1);
fsT=4;          % Normalized sampling rate, per bit (fs=sampling frequency,
% T=bit time =1 here)
Eb2=6;            % Energy per bit to spectral noise density ratio% (Eb/N0) for user 2, in dB
Eb2n=10^(Eb2/10);  % Eb/N0 for user 2, numeric
A2=sqrt(2*Eb2n/fsT);% Amplitude of user 2 MSK signal (Tb=1)
for NFR=1:1:30;             % Near-far ratio, in dB (Eb/N0)strong/(Eb/N0)weak
A1=A2*(10^((NFR/20)));% Amplitude of user 1 MSK signal
Eb1n=(A1^2)*fsT/2;  % Eb/N0 for user 1, numeric
Eb1=10*log10(Eb1n);  % Eb/N0 for user 1, in dB
Na=1;               % Na = number of antennas used; must be either 1 or 2
Eb1eff(NFR)=10*log10(Eb1n/(1+Eb2n));% Effective Eb/N0 for user 1,Eb1/(N0+Eb2)
Eb2eff(NFR)=10*log10(Eb2n/(1+Eb1n)); % Effective Eb/N0 for user 2,Eb2/(N0+Eb1)
\[ f_{D1} = 0.01; \quad \text{% Doppler frequency of strong (user 1) signal, \%relative to } T_b, \text{ i.e., } f_D = f_{Doppler} T_b \]
\[ \phi_1 = 2\pi \text{rand}(1,1); \quad \text{% Carrier phase of strong signal} \]
\[ t_{aus} = \text{rand}(1,1) \frac{1}{f_{Tb}}; \quad \text{% Sampling phase offset for strong signal 1, uniform between 0 and } T_s/N_s = T_s/2/f_{Tb} = 2T/2/f_{Tb} = 1/f_{Tb} \]
\[ t_{aus} = 0; \quad \text{Ae} = 0; \quad \text{% Percentage amplitude estimation error on strong signal, \%used to assess effect of imperfect amp estimation on IC} \]
\[ f_{1e} = 0; \quad \text{% Percentage frequency estimation error on strong signal} \]
\[ N = 400; \quad \text{% Number of bits used, must be EVEN. Generally, } 200 < N < 800. \]

\begin{verbatim}
if mod(N,2) ~= 0; N=N+1; end
Lv = f_{Tb} N; \quad \text{% Length of complex transmitted vectors v1 and v2}
Lv = Lv + f_{Tb}; \quad \text{% Extend vector by one bit, f_{Tb} samples}
LvA = 3*Lv; \quad \text{% Allow for asynchronous reception, as follows:}

t = 0:1/f_{Tb}:N-1/f_{Tb}; \quad \text{% Create time vector for plots}
\end{verbatim}

% Generate MSK via OFFSET QPSK, filtered with a sinusoidal pulse.
\[ u = \sin(\pi/T_s*\text{(0:1/f}_{Tb}:T_s-1/f_{Tb})+t_{aus}\pi/T_s); \quad \text{% Create sinusoidal pulse shape} \]
\[ N_{over} = 32; \]
\[ uV = \sin(\pi/2*(0:1/(N_{over}/2):2-1/(N_{over}/2))); \quad \text{% Create "template" pulse, N_{over} samples/T_s, to which estimate can be compared} \]
\[ b1 = \text{BIN01}(N,0.5); \]
\[ b2 = \text{BIN01}(N,0.5); \quad \text{% Generate random binary vectors} \]
\[ \text{bover1} = \text{OverN}(b1,f_{Tb}); \]
\[ \text{bover2} = \text{OverN}(b2,f_{Tb}); \quad \text{% Oversample} \]
\[ I1 = \text{zeros}(1,N*f_{Tb}+f_{Tb}); \quad \text{% Initialize I1 and Q1 components} \]
\[ Q1 = \text{zeros}(1,N*f_{Tb}+f_{Tb}); \]
\[ I2 = \text{zeros}(1,N*f_{Tb}+f_{Tb}); \]
\[ Q2 = \text{zeros}(1,N*f_{Tb}+f_{Tb}); \quad \text{% Initialize I2 and Q2 components} \]
\begin{verbatim}
for ii = 1:2:N; \quad \text{% Loop to create MSK baseband vectors}
I1((ii-1)*f_{Tb}+1:(ii+1)*f_{Tb}) = (-2*b1(ii)+1)*u; \quad \text{% '0' maps to 1; '%1' maps to -1}
Q1((ii)*f_{Tb}+1:(ii+2)*f_{Tb}) = (-2*b1(ii+1)+1)*u;
I2((ii-1)*f_{Tb}+1:(ii+1)*f_{Tb}) = (-2*b2(ii)+1)*u;
Q2((ii)*f_{Tb}+1:(ii+2)*f_{Tb}) = (-2*b2(ii+1)+1)*u;
end \end{verbatim}
% Generate Doppler envelope for strong signal
Adlc=cos(2*pi*fD1*[0:N*fsT+fsT-1]/fsT+phi1);
Adls=sin(2*pi*fD1*[0:N*fsT+fsT-1]/fsT+phi1);
% Generate baseband MSK signals, with Doppler envelope (so far Doppler only on strong signal)
v1=A1*(I1(1:N*fsT+fsT)-Q1(1:N*fsT+fsT));
v1=v1+j*A1*(-I1(1:N*fsT+fsT)-Q1(1:N*fsT+fsT));
v2=A2*(I2(1:N*fsT+fsT)-j*Q2(1:N*fsT+fsT));
% For burst Asynchronous, set bA=1
bA=0;
if bA == 1
    v1=[zeros(1,Lv) v1 zeros(1,Lv)]; % Lengthen signal 1 to 3*Lv for async
    tau=floor(rand(1,1)*(2*Lv+1)); % User 2 random delay, between 0 and 2*Lv samples
    v2=[zeros(1,tau) v2 zeros(1,2*Lv-tau)]; % User 2 signal, async
    n1=(randn(1,LvA)+j*randn(1,LvA))*1; % Baseband AWGN
    rA1=v1+v2+n1; % Baseband received signal
    rs1=rA1(Lv+1:2*Lv); % Extract signal 1 subsequence from r
else
    v2=cshift(v2,Randinteg(1,1,fsT-1));
    n1=randn(1,Lv)+j*randn(1,Lv);
    rA1=v1+v2+n1;
    rs1=rA1; % Extract signal 1 subsequence from r
    tau=0;
end
rAe=sqrt(sum(real(rs1).^2+imag(rs1).^2)/length(rs1)); % sqrt(rI^2 + rQ^2 ) amp estimate

I1hat=real(rs1(1:N*fsT)); Q1hat=imag(rs1(1:N*fsT+fsT));
% Organize samples by position within each symbol--get 2*fsT samples/symbol, so 2*fsT sub-vectors
Imat=reshape(I1hat,2*fsT,N/2); Qmat=reshape(Q1hat,2*fsT,N/2);
Im=Imat'; Qm=Qmat'; % Each column of Im (or Qm) has samples in same position
% uhatI and uhatQ are each vectors of 2*fsT samples, estimate of pulse sin(pi*(t+taus)/(2*T)),
t=k*fsT, k=0,1,...2*fsT-1
uhatI=2*sum(abs(Im))/N; uhatQ=2*sum(abs(Qm))/N;
% Next need to "extract" amplitude A1 from these averaged pulse samples;
uest=(uhatI+uhatQ)/2; % Averaging of BOTH I and Q samples together
uestS=overN(uest/A1,Nover/Ns); % Oversample estimate for determining delay estimate tauhat
(assumes "perfect" scaling with true A1)
uve=abs(uestS-uV); % Compare oversampled estimate with"template" pulse uV
[du,iu]=min(uev);               % Find index of minimum difference between estimate & template 
mdelay=mod(iu,Nover/Ns);        % NOTE: NOT CERTAIN OF ACCURACY of the "if" function here 
if mdelay == 0; 
    mdelay; 
    tauh=((Nover/Ns)-1)/Nover; 
else 
    tauh=(mod(iu,Nover/Ns)-1)/Nover; % tauh is quantized version of tauhat, in increments of 1/Nover 
end 
tauhat=Ts*tauh;                 % Final estimate of taus 
u2=sin(pi/Ts*(0:1/fsT:Ts-1/fsT)+tauhat*pi/Ts); 
A1hat=max(uest)/max(u2);

if BER1 == 1;                   % Compute user 1 Pb estimate if flag set 
    Pb1=0; Pb2=0;                    % Initialize Pb's 
    Ih11=zeros(1,N*fsT);     Qh11=zeros(1,N*fsT+fsT); % Initialize I1 and Q1 component estimates 
    Ih12=zeros(1,N*fsT);     Qh12=zeros(1,N*fsT+fsT); % Initialize I2 and Q2 component estimates 
    % Next, receiver correlation and detection for signal 1, assuming estimated carrier freq & phase, & 
    perfect symbol timing 
    for ir=1:2:N 
        Ih11((ir-1)*fsT+1:(ir+1)*fsT)=u.*real(rs1((ir-1)*fsT+1:(ir+1)*fsT))*sqrt(2); 
        Ihat11((ir+1)/2)=sign(sum(Ih11((ir-1)*fsT+1:(ir+1)*fsT)));  %I-channel estimate of signal 1, 
        hard decision 
        Ihat11((ir+1)/2)=sum(Ih11((ir-1)*fsT+1:(ir+1)*fsT)));  %I-channel estimate of signal 1, soft 
        decision 
        Qh11((ir*fsT+1:(ir+2)*fsT))=-u.*imag(rs1((ir*fsT+1:(ir+2)*fsT)))sqrt(2); 
        Qhat11((ir+1)/2)=sign(sum(Qh11((ir*fsT+1:(ir+2)*fsT)));%Q-channel estimate of signal 1, hard 
        decision 
        Qhat11((ir+1)/2)=sum(Qh11((ir*fsT+1:(ir+2)*fsT)));% Q-channel estimate of signal 1, soft 
        decision 
    end 
    bh11=LEAVE2(Ihat11,Qhat11);     % Interleave soft I and Q estimates for signal 1 
    bh1=(1-sign(bh11))/2;           % Hard decision estimates for signal 1 
    bh1H=sign(bh1);                % Hard bit decisions on signal 1 
    bhat1=(1-bh1H)/2;              % Convert +/-1 values to 0/1 values 
    er1=abs(b1-bhat1);            % Count bit errors, and compute Pb1 estimate 
    Pb1=sum(er1)/N;
end

% Next, IC: remodulate estimated user 1 bits and subtract from r, then %demodulate user 2, on each antenna separately

I1r1=zeros(1,Lv);  Q1r1=zeros(1,Lv);  % Initialize reconstructed I1 and %Q1 components, antenna 1
I1r2=zeros(1,Lv);  Q1r2=zeros(1,Lv);  % Initialize reconstructed I1 and %Q1 components, antenna 2

for ic=1:2:N;               % Loop to remodulate user 1 bit estimates
    I1r1((ic-1)*fsT+1:(ic+1)*fsT) = (-2*bh1(ic)+1)*u;
    Q1r1(ic*fsT+1:(ic+2)*fsT) = (-2*bh1(ic+1)+1)*u;
    I1r2((ic-1)*fsT+1:(ic+1)*fsT) = (-2*bhat1A2(ic)+1)*u;
    Q1r2(ic*fsT+1:(ic+2)*fsT) = (-2*bhat1A2(ic+1)+1)*u;
end

A1h=A1+((2*BIN01(1,0.5)-1)*Ae/100)*A1; % Introduce strong-signal amplitude error of Ae

% Reconstructed user 1 baseband MSK signal, using estimated amp & gain, %ant 1, with estimated Doppler
v1r1=A1hat*(I1r1(1:N*fsT+fsT)-Q1r1(1:N*fsT+fsT));
v1r1=v1r1+j*A1hat*(-I1r1(1:N*fsT+fsT)-Q1r1(1:N*fsT+fsT));
r1=rA1-v1r1;% IC the user 1 signal from the first-stage %received signal, ant 1
rICs1=r1;

for i2=1:2:N
    Ih21((i2-1)*fsT+1:(i2+1)*fsT) = u.*real(rICs1((i2-1)*fsT+1:(i2+1)*fsT))*sqrt(2);
    Ihat21((i2+1)/2)=sum(Ih21((i2-1)*fsT+1:(i2+1)*fsT));    % Antenna 1 I-channel estimate of signal 2, soft decision
    Qh21((i2+1)/2)=-u.*imag(rICs1(i2*fsT+1:(i2+2)*fsT))*sqrt(2);
    Qhat21((i2+1)/2)=sum(Qh21(i2*fsT+1:(i2+2)*fsT));        % Antenna 1 Q-channel estimate of signal 2, soft decision
end
bh21=LEAVE2(Ihat21,Qhat21);  % Interleave soft I and Q estimates for signal 2, antenna 1
bhat2A1=(1-sign(bh21))/2;    % Hard decision estimates for signal 2, antenna 1

bh2=bh21;  % Combine soft estimates of signal 2 from both antennas, then make hard decision.
Current combining method is MRC
bh2H=sign(bh2);      % Hard bit decisions on signal 2, after IC
bhat2=(1-bh2H)/2;    % Convert +/-1 values to 0/1 values
er2 = abs(b2 - bhat2);       % Count bit errors, and compute Pb2 estimate
Pb2(NFR) = sum(er2) / N;
NFR = NFR + 1;
end
Pb6 = Pb6 + Pb2;
end
avg6 = Pb6 / 300;
disp('the values of Eb2 in db are');
disp(avg6);

NFR = [1:1:30];
semilogy(NFR, avg6, 'r');
hold on;
grid on;
semilogy(NFR, avg5, 'r');
hold on;
semilogy(NFR, avg4, 'k');
hold on;
semilogy(NFR, avg3, 'k');
hold on;
semilogy(NFR, avg2);
hold on;
semilogy(NFR, avg, 'r');
grid on;
xlabel('NFR (db)-->');
ylabel('P_b_2 -->');
axis([0 30 0.001 1])
title('Plot of P_b_2 (weak signal) vs NFR ')
legend('P_b_2- using estimate of A_1'; 'P_b_2- using exact value of A_1')